www.archive.org
415.561.6767
415.840-0391 e-fax

Internet Archive
300 Funston Avenue
San Francisco, CA 94118

# AFFIDAVIT OF CHRISTOPHER BUTLER

1. I am the Office Manager at the Internet Archive, located in San Francisco, California. I make this declaration of my own personal knowledge.

2. The Internet Archive is a website that provides access to a digital library of Internet sites and other cultural artifacts in digital form. Like a paper library, we provide free access to researchers, historians, scholars, and the general public. The Internet Archive has partnered with and receives support from various institutions, including the Library of Congress.

3. The Internet Archive has created a service known as the Wayback Machine. The Wayback Machine makes it possible to surf more than 450 billion pages stored in the Internet Archive's web archive. Visitors to the Wayback Machine can search archives by URL (i.e., a website address). If archived records for a URL are available, the visitor will be presented with a list of available dates. The visitor may select one of those dates, and then begin surfing on an archived version of the Web. The links on the archived files, when served by the Wayback Machine, point to other archived files (whether HTML pages or images). If a visitor clicks on a link on an archived page, the Wayback Machine will serve the archived file with the closest available date to the page upon which the link appeared and was clicked.

4. The archived data made viewable and browseable by the Wayback Machine is compiled using software programs known as crawlers, which surf the Web and automatically store copies of web files, preserving these files as they exist at the point of time of capture.

5. The Internet Archive assigns a URL on its site to the archived files in the format http://web.archive.org/web/[Year in yyyy][Month in mm][Day in dd][Time code in hh:mm:ss]/[Archived URL]. Thus, the Internet Archive URL http://web.archive.org/web/19970126045828/http://www.archive.org/ would be the URL for the record of the Internet Archive home page HTML file (http://www.archive.org/) archived on January 26, 1997 at 4:58 a.m. and 28 seconds (1997/01/26 at 04:58:28). A web browser may be set such that a printout from it will display the URL of a web page in the printout's footer. The date assigned by the Internet Archive applies to the HTML file but not to image files linked therein. Thus images that appear on a page may not have been archived on the same date as the HTML file. Likewise, if a website is designed with "frames," the date assigned by the Internet Archive applies to the frameset as a whole, and not the individual pages within each frame.

6. Attached hereto as Exhibit A are true and accurate copies of printouts of the Internet Archive's records of the HTML files or PDF files for the URL and the dates specified in the footer of the printout (HTML) or attached coversheet (PDF).




7. Attached hereto as Exhibit B is a true and accurate electronic copies of the Internet Archive's records of the tar.gz files for the dates and URLs indicated in the following list and in the electronic folder in which each file for Exhibit B resides:

https://web.archive.org/web/20060112063540if_/http://www.snort.org:80/pub-bin/downloads.cgi/Download/vrt_pr/snortrules-pr-2.3.tar.gz

8. I declare under penalty of perjury that the foregoing is true and correct.

DATE: 5/8/19

Christopher Butler

CALIFORNIA JURAT

See Attached Document.

A notary public or other officer completing this certificate verifies only the identity of the individual who signed the document to which this certificate is attached, and not the truthfulness, accuracy, or validity of that document.

State of California
County of San Francisco

Subscribed and sworn to (or affirmed) before me on this

8 day of May, 2019, by

Christopher Butler,

proved to me on the basis of satisfactory evidence to be the person who appeared before me.

Signature: _____

LAUREL KARR
Notary Public - California
San Francisco County
Commission # 2172222
My Comm. Expires Nov 17, 2020

# Exhibit A

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 004

Skip to Navigation

SOURCEfire


- Customer Support
- Contact Us
- How To Buy
- Advisories
- Resources

- German
- French
- Japanese

Search this site:
Search

- Security Technologies
  - Sourcefire 3D System
    - Centralized Management
    - Virtual Security
    - Intrusion Prevention System (IPS)
    - Intrusion Detection System (IDS)
    - SSL Inspection
    - Network Awareness
    - User Awareness
    - Netflow Analysis
  - Sourcefire VRT
  - Snort
  - ClamAV
- Industry/Compliance
- Services
- Partners
- About Us


Latest Advisory:
December 14th, 2010
Microsoft Tuesday Coverage for December 2010

Home • Security Technologies • Sourcefire 3D System

# SSL Inspection

Strengthen Security with Secure Sockets Layer (SSL) Inspection

**SSL-encrypted Traffic—An Easy Vehicle for Cybersecurity Attacks**



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 005
Keysight, EX-1045
Page 5



SSL-encrypted traffic is exploding due to the enterprise-wide usage of cloud computing, secure e-commerce, Web 2.0 applications, email, and VPNs. Surveys show 25-35% of enterprise traffic is SSL-encrypted, and this number is up to 70% for select verticals. If not managed properly, SSL can leave a hole in any enterprise security architecture. Existing approaches to SSL-encrypted traffic range from passing everything to blocking everything. In some cases, companies deploy host-based IPS systems or install proxy SSL solutions, which can effectively inspect SSL but suffer from bottleneck issues and reduced network performance.

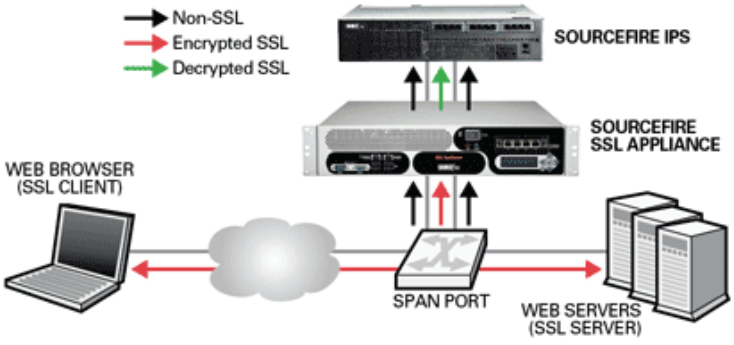
**Decrypts SSL Traffic at 1Gbps Line Rate**

The Sourcefire SSL Appliance decrypts SSL traffic and sends it to existing security and network appliances via dedicated gigabit Ethernet links. This enables existing IPS appliances to identify risks normally hidden by SSL such as regulatory compliance violations, viruses, malware, data loss, and intrusion attempts. Once the SSL traffic has been inspected and approved, the SSL Appliance places the SSL-encrypted traffic back on the network for its final destination—all with minimal latency and without altering SSL packets. Operates Transparently on Network The SSL Appliance is deployed as a transparent proxy and detects SSL sessions on all ports, not just the traditional port 443. It can run as a "bump-in-the-wire" and does not require network configuration, IP addressing or topology changes, or modification to client IP and web browser configurations. Further, transparent SSL proxies see all network traffic, not just SSL, and have the ability to cut-through non-SSL flows.

**Supports Passive and Inline Configurations**

The SSL Appliance supports both passive and inline configurations. When deployed passively, it sends traffic to a Sourcefire IPS also running in passive mode. Passive deployment is most useful for gaining full visibility into network traffic and what vulnerabilities may be exploited. The SSL Appliance can also be deployed inline as a "bump-in-the-wire" and operate with an IPS running in either passive or inline mode. When both the SSL Appliance and the IPS are deployed inline, they can block malicious exploit traffic. All Sourcefire SSL Appliances ship with fail-open 4-port 1G copper or fiber interfaces

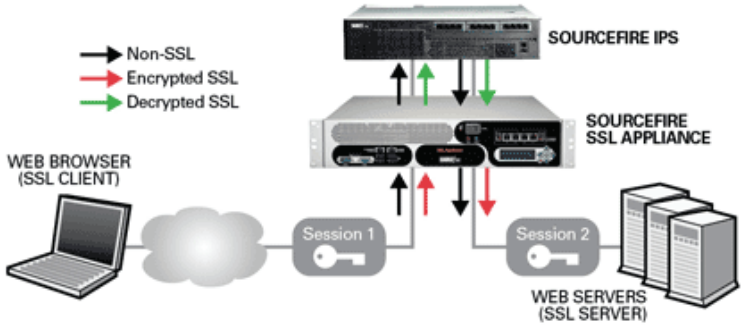
**Passive IDS Configuration**




**Inline IPS Configuration**



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 006
Keysight, EX-1045
Page 6




The SSL Appliance is versatile enough to inspect SSL traffic in both inbound and outbound configurations. With inbound SSL inspection, the appliance inspects traffic destined for an enterprise's web servers hosting SSL applications. With outbound SSL inspection, the appliance inspects SSL application traffic destined outside of the enterprise, such as Google Gmail traffic.

## Security Technologies

- Sourcefire 3D System
  - Centralized Management
  - Virtual Security
  - Intrusion Prevention System (IPS)
  - Intrusion Detection System (IDS)
  - SSL Inspection
  - Network Awareness
  - User Awareness
  - Netflow Analysis
- Sourcefire VRT
- Snort
- ClamAV

## White Papers

SANS Calculating TCO with Adaptive IPS White Paper
Sourcefire Vulnerability Research Team (VRT) White Paper

View All

## Buy Sourcefire Products

## Related Resources

- Sourcefire SSL Appliance Fact Sheet
- Sourcefire SSL Appliance Technology Brief
- Sourcefire IPS Fact Sheet



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 007
Keysight, EX-1045
Page 7

# Archived Webinars

HIPAA Compliance and IPS with Sisters of Charity Providence Hospitals
View Webcast
Exploring Passive Analysis and White Lists with XanGo
View Webcast
PCI Compliance and IPS with BankersBankCard Services (BBCS/TIB)
View Webcast

View All





- Snort Home
- Cybersecurity Home
- IPS
- Snort
- Advisories/Rules
- Newsroom
- Events
- Investors
- Careers
- Legal

- Facebook
- Twitter
- LinkedIn
- YouTube


©2010 Sourcefire, Inc. Website Design by Qorvis

# Cryptology ePrint Archive: Report 2006/136

**A Challenging but Feasible Blockwise-Adaptive Chosen-Plaintext Attack on SSL**

*Gregory V. Bard*

**Abstract.** This paper introduces a chosen-plaintext vulnerability in the Secure Sockets Layer (SSL) and Trasport Layer Security (TLS) protocols which enables recovery of low entropy strings such as can be guessed from a likely set of 2--1000 options. SSL and TLS are widely used for securing communication over the Internet. When utilizing block ciphers for encryption, the SSL and TLS standards mandate the use of the cipher block chaining (CBC) mode of encryption which requires an initialization vector (IV) in order to encrypt. Although the first IV used by SSL is a (pseudo)random string which is generated and shared during the initial handshake phase, subsequent IVs used by SSL are chosen in a deterministic, predictable pattern; in particular, the IV of a message is taken to be the final ciphertext block of the immediately-preceding message, and is therefore known to the adversary.

The one-channel nature of web proxies, anonymizers or Virtual Private Networks (VPNs), results in all Internet traffic from one machine traveling over the same SSL channel. We show this provides a feasible ``point of entry" for this attack. Moreover, we show that the location of target data among block boundaries can have a profound impact on the number of guesses required to recover that data, especially in the low-entropy case.

The attack in this paper is an application of the blockwise-adaptive chosen-plaintext attack paradigm, and is the only feasible attack to use this paradigm with a reasonable probability of success. The attack will work for all versions of SSL, and TLS version 1.0. This vulnerability and others are closed in TLS 1.1 (which is still in draft status) and OpenSSL after 0.9.6d. It is hoped this paper will encourage the deprecation of SSL and speed the adoption of OpenSSL or TLS 1.1/1.2 when they are finially released.

**Category / Keywords.** implementation / Blockwise Adaptive, Chosen Plaintext Attack (CPA), Secure Sockets Layer (SSL), Transport Layer Security (TLS), Cryptanalysis, HTTP-proxy, Initialization Vectors (IV), Cipher Block Chaining (CBC), Virtual Private Networks (VPN).

**Publication Info.** Not yet published. See E-print 2004/111 which is similar in theory but very different in the actual attack.

**Date:** received 5 Apr 2006, last revised 17 Apr 2006

**Contact author:** gregory bard at ieee org

**Available formats:** PDF | BibTeX Citation

**Comment.** Changes relating to the directions of traffic of the adversary and target. Also some formatting changes.

---

[ Cryptology ePrint archive ]



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 009
**Keysight, EX-1045**
**Page 9**

Internet Engineering Task Force (IETF)
Request for Comments: 6176
Updates: 2246, 4346, 5246
Category: Standards Track
ISSN: 2070-1721

S. Turner
IECA
T. Polk
NIST

March 2011

# Prohibiting Secure Sockets Layer (SSL) Version 2.0

## Abstract

This document requires that when Transport Layer Security (TLS) clients and servers establish connections, they never negotiate the use of Secure Sockets Layer (SSL) version 2.0. This document updates the backward compatibility sections found in the Transport Layer Security (TLS).

## Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6176.

## Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.



https://web.archive.org/web/20110318184421/https://tools.ietf.org/html/rfc6176

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 010
Keysight, EX-1045
Page 10

## 1. Introduction

Many protocols specified in the IETF rely on Transport Layer Security (TLS) [TLS1.0][TLS1.1][TLS1.2] for security services. This is a good thing, but some TLS clients and servers also support negotiating the use of Secure Sockets Layer (SSL) version 2.0 [SSL2]; however, this version does not provide a sufficiently high level of security. SSL version 2.0 has known deficiencies. This document describes those deficiencies, and it requires that TLS clients and servers never negotiate the use of SSL version 2.0.

RFC 4346 [TLS1.1], and later RFC 5246 [TLS1.2], explicitly warned implementers that the "ability to send version 2.0 CLIENT-HELLO messages will be phased out with all due haste". This document accomplishes this by updating the backward compatibility sections found in TLS [TLS1.0][TLS1.1][TLS1.2].

### 1.1. Requirements Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

## 2. SSL 2.0 Deficiencies

SSL version 2.0 [SSL2] deficiencies include the following:

- Message authentication uses MD5 [MD5]. Most security-aware users have already moved away from any use of MD5 [RFC6151].
- Handshake messages are not protected. This permits a man-in-the-middle to trick the client into picking a weaker cipher suite than it would normally choose.
- Message integrity and message encryption use the same key, which is a problem if the client and server negotiate a weak encryption algorithm.
- Sessions can be easily terminated. A man-in-the-middle can easily insert a TCP FIN to close the session, and the peer is unable to determine whether or not it was a legitimate end of the session.



https://web.archive.org/web/20110318184421/https://tools.ietf.org/html/rfc6176

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 011
Keysight, EX-1045
Page 11

## 3. Changes to TLS

Because of the deficiencies noted in the previous section:

- TLS clients MUST NOT send the SSL version 2.0 compatible CLIENT-HELLO message format. Clients MUST NOT send any ClientHello message that specifies a protocol version less than { 0x03, 0x00 }. As previously stated by the definitions of all previous versions of TLS, the client SHOULD specify the highest protocol version it supports.

- TLS servers MAY continue to accept ClientHello messages in the version 2 CLIENT-HELLO format as specified in RFC 5246 [TLS1.2], Appendix E.2. Note that this does not contradict the prohibition against actually negotiating the use of SSL 2.0.

- TLS servers MUST NOT reply with an SSL 2.0 SERVER-HELLO with a protocol version that is less than { 0x03, 0x00 } and instead MUST abort the connection, i.e., when the highest protocol version offered by the client is { 0x02, 0x00 }, the TLS connection will be refused.

Note that the number of servers that support this above-mentioned "MAY accept" implementation option is declining, and the SSL 2.0 CLIENT-HELLO precludes the use of TLS protocol enhancements that require TLS extensions. TLS extensions can only be sent as part of an (Extended) ClientHello handshake message.

## 4. Security Considerations

This entire document is about security considerations.

## 5. Acknowledgements

The idea for this document was inspired by discussions between Peter Saint Andre, Simon Josefsson, and others on the Extensible Messaging and Presence Protocol (XMPP) mailing list.

We would also like to thank Michael D'Errico, Paul Hoffman, Nikos Mavrogiannopoulos, Tom Petch, Yngve Pettersen, Marsh Ray, Martin Rex, Yaron Sheffer, and Glen Zorn for their reviews and comments.



https://web.archive.org/web/20110318184421/https://tools.ietf.org/html/rfc6176

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 012
**Keysight, EX-1045**
**Page 12**

## 6. References

### 6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[TLS1.0] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[TLS1.1] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

[TLS1.2] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

### 6.2. Informative References

[MD5] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

[SSL2] Hickman, Kipp, "The SSL Protocol", Netscape Communications Corp., Feb 9, 1995.

[RFC6151] Turner, S. and L. Chen, "Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms", RFC 6151, March 2011.

Authors' Addresses

Sean Turner
IECA, Inc.
3057 Nutley Street, Suite 106
Fairfax, VA 22031
USA

EMail: turners@ieca.com

Tim Polk
National Institute of Standards and Technology
100 Bureau Drive, Mail Stop 8930
Gaithersburg, MD 20899-8930
USA

EMail: tim.polk@nist.gov



https://web.archive.org/web/20110318184421/https://tools.ietf.org/html/rfc6176

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 013
Keysight, EX-1045
Page 13

https://web.archive.org/web/20110318184421/https://tools.ietf.org/html/rfc6176

NDM
877-489-1113

SOURCEfire®
AUTHORIZED DISTRIBUTOR

HOME BUY PRODUCTS HARDWARE & SOFTWARE TECHNOLOGY SOURCEFIRE 3D SYSTEM INDUSTRIES RESOURCES QUOTE LOWEST PRICE search...

# PRODUCT DATA SHEETS

**Sourcefire 3D System and Citrix NetScaler**

Sourcefire IPS™ works in combination with Citrix NetScaler® Web Application Delivery Appliance for a layered approach to security.

**NSS Labs Network IPS Group Test Executive Summary**

Executive summary of the NSS Labs network intrusion prevention group test done in December 2009. The Sourcefire 3D4500 Sensor, part of the Sourcefire 3D System intrusion prevention family, achieved a "Recommend" rating.

**Sourcefire 3D System and Solera Networks Fact Sheet**

Sourcefire and Solera Networks have partnered together to provide you with full context for every attack on your network so you can swiftly respond to any security incident and better prepare for future attack attempts.

**Sourcefire IPS Fact Sheet**

Learn about the power, flexibility, precision, scalability, and performance of Sourcefire IPS™.

**Sourcefire Virtual Appliances Fact Sheet**

The Sourcefire Virtual 3D™ Sensor and Sourcefire Virtual Defense Center™ enable users to deploy Sourcefire's leading cybersecurity solutions within their virtual environments, increasing protection for both physical and virtual assets.

**Sourcefire SSL Appliance Fact Sheet**

Strengthen security with secure sockets layer (SSL) inspection. The Sourcefire SSL Appliance decrypts SSL-encrypted traffic at 1Gbps line rate inspection.

**Sourcefire 3D9900 Sensor Fact Sheet**

A detailed look at the features, specifications, and benefits of Sourcefire's 10Gbps IPS appliance.

**Sourcefire Defense Center Fact Sheet**

The "nerve center" of the Sourcefire 3D® System, Sourcefire Defense Center® correlates attacks with real-time network and user intelligence and centrally manages network security and operational functions, including event monitoring, incident prioritization, forensic analysis, and reporting.

**Sourcefire RNA Fact Sheet**

The innovative, passive-sensing Sourcefire RNA® technology transforms how organizations defend their networks, providing network and security administrators with unprecedented real-time network visibility.

**Sourcefire 3D System and SSL Appliance Specifications**

Specification matrix covering the full range of appliances within the Sourcefire 3D® System family (Sourcefire 3D Sensors and Defense Centers) and the Sourcefire SSL Appliance.

https://web.archive.org/web/20111028104123/http://www.ndm.net:80/sourcefirestore/Product-Data-Sheets/product-data-sheets

**Sourcefire NBA Overview**

Understand the benefits of the latest generation of layered defense with Network Behavior Analysis (NBA), a key component of Sourcefire's solution.

**Sourcefire RUA Fact Sheet**

Learn about the benefits of integrated real-time user awareness with Sourcefire RUA®.

**ICSA Labs Network IPS Certification Testing Report**

ICSA Labs Network IPS Certification Testing Report for the complete line of Sourcefire 3D® Sensors.

**Sourcefire 3D System and QualysGuard Integration Fact Sheet**

Sourcefire partners with Qualys to provide contextual network detail and more effective threat impact analysis, via the integration of QualysGuard vulnerability data into the Sourcefire 3D System.

**Sourcefire 3D System and Bradford Networks' Access Solutions Fact Sheet**

Sourcefire and Bradford Networks have partnered to provide integration of Sourcefire IPS and Sourcefire RNA with Bradford Networks' Campus Manager and NAC Director™ for automated security remediation.

**Sourcefire 3D for Crossbeam - Miercom Lab Testing Summary Report**

Miercom's test results of the Crossbeam-Sourcefire solution for scalability, throughput performance, high availability, and multi-application capabilities.

**Sourcefire 3D System for Crossbeam Fact Sheet**

Fact sheet addressing the benefits of choosing the Sourcefire 3D System for Crossbeam including: Best-in-class integrated security, high availability, investment protection, performance/scalability, and ROI savings.

Live Chat

**TOP IPS SOLUTIONS**

RESOURCES MENU

Vertical Industries

Compliance Solutions

Product Data Sheets

Technology Briefs

Case Studies

White Papers

Quote Request

Buy

## DEFENSE CENTER

*Sourcefire DC500 - Manage 3 Sensors*
*Sourcefire DC750 - Manage 10 Sensors*
*Sourcefire DC1000 - Manage 25 Sensors*
*Sourcefire DC1500 - Manage 35 Sensors*
*Sourcefire DC3000 - Manage 100 Sensors*
*Sourcefire DC 3500 - Manage 150 Sensors*

## SOURCEFIRE 3D SENSORS

*Sourcefire 3D500 - 5 Mbps*
*Sourcefire 3D1000 - 45 Mbps*
*Sourcefire 3D2000 - 100 Mbps*
*Sourcefire 3D2100 - 250 Mbps*
*Sourcefire 3D2500 - 500 Mbps*
*Sourcefire 3D3500 - 1 Gbps*
*Sourcefire 3D4500 - 2 Gbps*
*Sourcefire 3D6500 - 4 Gbps*
*Sourcefire 3D8140 - 6 Gbps*
*Sourcefire 3D8250 - 10 Gbps*
*Sourcefire 3D8260 - 20 Gbps*
*Sourcefire 3D9900 - 10 Gbps*

## SOURCEFIRE IPSX

*Sourcefire DC750x - Manage 10 Sensors*
*Sourcefire IPSx250 Sensor - 250 Mbps*
*Sourcefire IPSx500 Sensor - 500 Mbps*
*Sourcefire IPSx1000 Sensor - 1 Gbps*



Copyright © 1994-2011 Sourcefire Authorize Partner - website operated by www.NDM.net







https://web.archive.org/web/20111028104123/http://www.ndm.net:80/sourcefirestore/Product-Data-Sheets/product-data-sheets

# An Overview of SSL (version 2)

**Adam Shostack, May 1995**

*SSL version 3 is in the draft stage. I believe that SSL version 2 is in Netscape Nevigator version 2.*

## A. Introduction to, and history of, SSL

SSL, Secure Sockets Layer, is a protocol designed and implemented by Netscape Communications. Netscape claims it is designed to work, as the name implies, at the socket layer, to protect any higher level protocol built on sockets, such as telnet, ftp, or HTTP. As such, it is ignorant of the details of higher level protocols, and what is being transported. A free reference version of SSL, SSLRef, is available from Netscape. Many of the functions provided by SSL are part of the newly defined IPv6.

SSL provides for encryption of a session, authentication of a server, and optionally a client, and message authentication. The SSL Handshake Protocol and the application protocol both operate on top of the SSL Record Protocol, a simple means of encapsulating authentication information. SSL-Record Layer works on TCP or some other reliable transport mechanism. Session establishment takes from 5 to 8 messages, depending on options used. SSL relies on the existence of a key certification mechanism for the authentication of a server. SSL does not provide for renegotiation of keys within a session. (This is not a problem in HTTP, but might be with other protocols.) A multitude of ciphers and secure hashes are supported, including some explicitly weakened to comply with export restrictions.

## B. The SSL protocols

### 1. SSL Record Protocol

All SSL protocol messages move in records of up to 32,767 bytes. Each message has a header of either 2 or 3 bytes. The headers include a security escape function, a flag to indicate the existence of padding, and the length of the message (and possibly the padding.) A two byte header has no padding, a three byte header includes some padding.

```
        The meaning of bits is as follows:

 1 2 3 4 5 6 7 8
+----------------+-----------------+----------------+
|# S Length      |  Length         | Padding length |
+----------------+-----------------+----------------+

# is the number of bytes in the header
        0 indicates a 3 byte header, max length 32,767 bytes.
        1 indicates a 2 byte header, max length 16,383 bytes S
indicates the presence of a security escape, although none are
currently implemented. (Several suggestions for security
escapes are in the weaknesses section.)
```

There is no version information within the SSL record header, although it is available in the handshake.

Within a record, there are three components: MAC-DATA, Actual-data, and Padding-data. MAC is the Message Authentication Code, Actual-data is the actual data being sent, and Padding-data is padding.

The MAC-DATA is a hash of a key, the data, padding, and a sequence number. The hash is chosen based on the cipher-choice. The key used is the (sender)-write-key, which is the same key as the (receiver)-read-key. When cipher-choice is not defined, there is no mac-data or padding-data.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 018
**Keysight, EX-1045**
**Page 18**

Padding is used to ensure that the data is a multiple of the block size when a block cipher is used. Padding data is always discarded after the MAC has been calculated.

Sequence numbers are unsigned 32 bit integers incremented with each message sent. Sequence numbers wrap to zero after 0xFFFFFFFF. Each dialog direction has its own sequence number.

Failure to authenticate, decrypt, or otherwise get correct answers in a crytpographic operation result in I/O errors, and a close of connection.

#### 2. SSL Handshake Protocol

A handshake occurs when a machine attempts to use a SSL connection. There are three basic types of handshaking. The first is when no session exists "recently". Recently is not explicitly defined, but suggested to be under 100 seconds (SSL, C.8). The second is when the a set of session identifiers still exists, the third is when client authentication is desired.

For the initial connection, when a client wishes to establish a secure connection, it sends a CLIENT-HELLO message, including a challenge, along with information on the cryptographic systems it is willing or able to support. The server responds with a SERVER-HELLO message, which is connection id, its key certificate, and information about the cryptosystems it supports. The client is responsible for choosing a cryptosystem it shares with the server.

The client then verifies the server's public key, and responds with a CLIENT-MASTER-KEY message, which is a randomly generated master key, encrypted or partially encrypted with the servers public key. The client then sends a CLIENT-FINISHED message. This includes the connection-id, encrypted with the client-write-key. (All these keys are explained separately, in the next section.) The server then sends a SERVER-VERIFY, verifying its identity by responding with the challenge, encrypted with the server write key. The server got its server-write-key sent to it by the client, encrypted with the server's public key. The server thus must have the appropriate private key to decrypt the CLIENT-MASTER-KEY message, thus obtaining the master-key, from which it can produce the server-write-key.

When a session identifier is found by the client, a slightly different initialization takes place. It is initially distinguishable by the CLIENT-HELLO message including a session-id. The SERVER-HELLO response includes a set "session_id_hit" bit if the session-id is found. The client then sends an encrypted connection-id for this session in the CLIENT-FINISH message. The server responds with a SERVER-VERIFY and a SERVER-FINISH message, which are close to identical to those used in the "No session identifier case." The only difference is that the SERVER-FINISH message contains a session-id instead of new-session-id.

If client authentication is in use, then the server must at some point, send a REQUEST-CERTIFICATE message, which contains a challenge (called challenge') and the means of authentication desired. The client responds with a CLIENT-CERTIFICATE message, which includes the client certificate's type, the certificate itself, and a bunch of response data. The server then sends a SERVER-FINISH message.

To summarize those messages, they are:

```
 2.2.1 Assuming no session-identifier

          client-hello          C -> S: challenge, cipher_specs
          server-hello          S -> C: connection-id,server_certificate,cipher_specs
          client-master-key     C -> S: {master_key} server_public_key
          client-finish         C -> S: {connection-id} client_write_key
          server-verify         S -> C: {challenge} server_write_key
          server-finish         S -> C: {new_session_id} server_write_key

2.2.2 Assuming a session-identifier was found by both client & server
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 019
Keysight, EX-1045
Page 19

```
          client-hello          C -> S: challenge, session_id, cipher_specs
          server-hello          S -> C: connection-id, session_id_hit
          client-finish         C -> S: {connection-id}client_write_key
          server-verify         S -> C: {challenge}server_write_key
          server-finish         S -> C: {session_id}server_write_key

2.2.3 Assuming a session-identifier was used and client
          authentication is used

          client-hello          C -> S: challenge, session_id, cipher_specs
          server-hello          S -> C: connection-id, session_id_hit
          client-finish         C -> S: {connection-id}client_write_key
          server-verify                 S -> C: {challenge}server_write_key
          request-certificate   S -> C: {auth_type,challenge'}server_write_key
          client-certificate    C -> S: {cert_type,client_cert, response_data}client_write_key
          server-finish         S -> C: {session_id}server_write_key
```

In this last exchange, the response_data is a function of the auth_type. (These tables are taken verbatim from the SSL Specification, Section numbers as indicated.)

There are some errors defined for the handshake. They are NO-CIPHER-ERROR, client NO-CERTIFICATE-ERROR, BAD-CERTIFICATE-ERROR, and UNSUPPORTED_CERTIFICATE_TYPE. NO-CIPHER occurs when the client and server can't agree on a cipher. (It is not clear why SSL doesn't require support for some simple cipher, such as RC4-40.) The other errors are recoverable if the client is the one with the bad certificate.

#### 3. Keys used in SSL

There are a number of keys used over the course of a conversation. There is the server's public key, a master key, a client-read-key and a client-write-key. (The standard uses the term server-write-key as another name for client-read-key, and server-read-key as another name for client-write-key.)

Client-write-key and client-read-key are derived via a secure hash from the master key, an ordinal character, the challenge, and connection-id. Of this input, only the master key is sent encrypted (with the server's public key.) The master key is reused across sessions, while the read- & write- keys are generated anew for each session.

Two or more sets of key material are generated, based on how many bits of key are needed. Hash tends to be MD5, but is not required to be so by the spec. It is MD5 for all cipher types currently defined.

key-material-N = hash (master key, "N", challenge, connection-id) Keys are taken from the output of these hashes. Because of the design of secure hashes, changing 'N' will result in the output of these hashes being very different.

#### 4. SSL Application Protocol

Once the handshaking is complete, the application protocol begins to operate. The SSL spec is not clear at what point the SSL connection is considered to be done with a connection, nor what to do with the keys at that point. There is an implication that the session is done when the TCP connection is torn down, and keys should be kept for roughly 100 seconds after that, but this is not explicitly defined.

#### 5. Other details

SSL uses a scheme of https to reference documents available under HTTP with SSL. The https has a IANA reserved number of 443. SSL supports the RC2 & RC4 with either 128 bits or 40 bits of secret key information, as well as DES, 3 key 3DES, and IDEA. Details of all are covered in Schneier.

### C. The threats



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 020
Keysight, EX-1045
Page 20

As long as they confine themselves to playing by the rules established by Netscape, lookers can do little harm. There's little a looker with minimal resources can do against the deployed crypto systems used in SSL. Hacking one or more of the hosts or key certification authorities would seem to be a better option.

Insiders, especially those around the top of the key certification hierarchy, have the potential to do quite a bit of harm by creating false signatures on keys. Few of these attacks will occur in a vengeful manner; they require time and foresight to enact, and are probably the domain of the malicious employee. (This assumes that employees who become vengeful do so at about the time they leave a firm.)

A criminal organization could, depending on its resources, possibly target an employee at a key certification organization. However, a more useful option might be to buy a cheap PC, and have it attempt brute force RC4 keys. It is estimated that a pentium based PC should be able to crack a 40 bit RC4 key in a month or several months using brute force. The manipulations used on the master key may increase the cost of tha attack, but probably not by orders of magnitude. If a PC costs $1500, then breaking 12 keys a year leads to a cost that could be as low as $125 per stolen card number. While this seems like a high price, the credit card numbers are acquired in a nearly risk free manner of sniffing an ethernet. In addition, that time will drop with the introduction of faster hardware.

**D. Protections**

SSL explicitly examines a number of attacks that can be made against it, including brute force, clear text cryptanalysis, replay, and man in the middle. It does seem to protect well against those forms of attack.

SSL is designed to protect at the network layer. This means it is not designed to, and does not, protect you from host breakins. To protect against host breakins, a Tripwire-like package should be integrated with the HTTPd. Tripwire uses cryptographically strong hashes to ensure documents have not been changed from some reference version. The database is usually stored on a physically write protected media like a floppy (Kim).

**E. Weaknesses**

SSL, being a low level protocol, does little to protect you once your host is compromised. Also, once a key in a certificate is compromised, it can remain compromised, as there is no mechanism in place for consulting the root of a CA to confirm the key you are using has not been revoked. The keys however do include expiration dates. Climbing to the root CA is not a commonplace step, but a mechanism should be available to do so, for high value transactions. Out of band key repudiation is also desirable. Today, trusted key certifiers are compiled into the Netscape binary.

The use of RC4 is troublesome, albeit understandable. RC4 is a newly published cipher, and although it was designed by the very competent Ron Rivest, it has not been subjected to the same kind of intense professional scrutiny that DES and IDEA have undergone. The decision to use RC4-40 is driven by the fact that it gets automatic export approval from the State Department.

There are also a number of areas in the design of SSL as it stands that could become exploitable problems. None suggest an immediate means of attack, but are things which could be modified for possible added surety.

Handshaking protocol: The challenge data, sent in the CLIENT-HELLO message, could in some types of handshakes, be sent later, encrypted. The data used in generating the session keys could include more data not sent in the clear. The means of generating the master key should be better specified, probably with reference to RFC-1750.

Record protocol: Bad MAC-data should not terminate a connection, it should cause a repeat-request message. There are few attacks that will get anything from having the same data resent, and closing the connection on a bad message opens avenues for denial of service attacks. Sequence numbers should be randomly initialized.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 021
**Keysight, EX-1045**
**Page 21**

There are quite a few non-obvious attacks on sequence numbers (in IP, and NFS); it can't hurt to start with a non-predictable number.

There should be a way for one or both sides to demand renegotiation of keys. Perhaps this could be implemented as a security escape. This is not needed for HTTP connection security, since the connections are very short lived, but if SSL is used, as the authors suggest, for telnet or FTP, the sessions could last substantially longer.



Previous Home Next



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 022
**Keysight, EX-1045**
**Page 22**

RFC: 791

# INTERNET PROTOCOL

## DARPA INTERNET PROGRAM

## PROTOCOL SPECIFICATION

September 1981

**prepared for**

Defense Advanced Research Projects Agency
Information Processing Techniques Office
1400 Wilson Boulevard
Arlington, Virginia 22209

by

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, California 90291

https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

TABLE OF CONTENTS

PREFACE ..... iii

1. INTRODUCTION ..... 1

1.1 Motivation ..... 1
1.2 Scope ..... 1
1.3 Interfaces ..... 1
1.4 Operation ..... 2

2. OVERVIEW ..... 5

2.1 Relation to Other Protocols ..... 9
2.2 Model of Operation ..... 5
2.3 Function Description ..... 7
2.4 Gateways ..... 9

3. SPECIFICATION ..... 11

3.1 Internet Header Format ..... 11
3.2 Discussion ..... 23
3.3 Interfaces ..... 31

APPENDIX A: Examples & Scenarios ..... 34
APPENDIX B: Data Transmission Order ..... 39

GLOSSARY ..... 41

REFERENCES ..... 45



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 025
Keysight, EX-1045
Page 25

https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 026
Keysight, EX-1045
Page 26

## PREFACE

This document specifies the DoD Standard Internet Protocol. This document is based on six earlier editions of the ARPA Internet Protocol Specification, and the present text draws heavily from them. There have been many contributors to this work both in terms of concepts and in terms of text. This edition revises aspects of addressing, error handling, option codes, and the security, precedence, compartments, and handling restriction features of the internet protocol.

Jon Postel

Editor



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 027
Keysight, EX-1045
Page 27

September 1981

RFC: 791
Replaces: RFC 760
IENs 128, 123, 111,
80, 54, 44, 41, 28, 26

INTERNET PROTOCOL

DARPA INTERNET PROGRAM
PROTOCOL SPECIFICATION

# 1. INTRODUCTION

## 1.1. Motivation

The Internet Protocol is designed for use in interconnected systems of packet-switched computer communication networks. Such a system has been called a "catenet" [1]. The internet protocol provides for transmitting blocks of data called datagrams from sources to destinations, where sources and destinations are hosts identified by fixed length addresses. The internet protocol also provides for fragmentation and reassembly of long datagrams, if necessary, for transmission through "small packet" networks.

## 1.2. Scope

The internet protocol is specifically limited in scope to provide the functions necessary to deliver a package of bits (an internet datagram) from a source to a destination over an interconnected system of networks. There are no mechanisms to augment end-to-end data reliability, flow control, sequencing, or other services commonly found in host-to-host protocols. The internet protocol can capitalize on the services of its supporting networks to provide various types and qualities of service.

## 1.3. Interfaces

This protocol is called on by host-to-host protocols in an internet environment. This protocol calls on local network protocols to carry the internet datagram to the next gateway or destination host.

For example, a TCP module would call on the internet module to take a TCP segment (including the TCP header and user data) as the data portion of an internet datagram. The TCP module would provide the addresses and other parameters in the internet header to the internet module as arguments of the call. The internet module would then create an internet datagram and call on the local network interface to transmit the internet datagram.

In the ARPANET case, for example, the internet module would call on a



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 028
Keysight, EX-1045
Page 28

local net module which would add the 1822 leader [2] to the internet datagram creating an ARPANET message to transmit to the IMP. The ARPANET address would be derived from the internet address by the local network interface and would be the address of some host in the ARPANET, that host might be a gateway to other networks.

### 1.4. Operation

The internet protocol implements two basic functions: addressing and fragmentation.

The internet modules use the addresses carried in the internet header to transmit internet datagrams toward their destinations. The selection of a path for transmission is called routing.

The internet modules use fields in the internet header to fragment and reassemble internet datagrams when necessary for transmission through "small packet" networks.

The model of operation is that an internet module resides in each host engaged in internet communication and in each gateway that interconnects networks. These modules share common rules for interpreting address fields and for fragmenting and assembling internet datagrams. In addition, these modules (especially in gateways) have procedures for making routing decisions and other functions.

The internet protocol treats each internet datagram as an independent entity unrelated to any other internet datagram. There are no connections or logical circuits (virtual or otherwise).

The internet protocol uses four key mechanisms in providing its service: Type of Service, Time to Live, Options, and Header Checksum.

The Type of Service is used to indicate the quality of the service desired. The type of service is an abstract or generalized set of parameters which characterize the service choices provided in the networks that make up the internet. This type of service indication is to be used by gateways to select the actual transmission parameters for a particular network, the network to be used for the next hop, or the next gateway when routing an internet datagram.

The Time to Live is an indication of an upper bound on the lifetime of an internet datagram. It is set by the sender of the datagram and reduced at the points along the route where it is processed. If the time to live reaches zero before the internet datagram reaches its destination, the internet datagram is destroyed. The time to live can be thought of as a self destruct time limit.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 029
Keysight, EX-1045
Page 29

The Options provide for control functions needed or useful in some situations but unnecessary for the most common communications. The options include provisions for timestamps, security, and special routing.

The Header Checksum provides a verification that the information used in processing internet datagram has been transmitted correctly. The data may contain errors. If the header checksum fails, the internet datagram is discarded at once by the entity which detects the error.

The internet protocol does not provide a reliable communication facility. There are no acknowledgments either end-to-end or hop-by-hop. There is no error control for data, only a header checksum. There are no retransmissions. There is no flow control.

Errors detected may be reported via the Internet Control Message Protocol (ICMP) [3] which is implemented in the internet protocol module.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 030
**Keysight, EX-1045**
**Page 30**

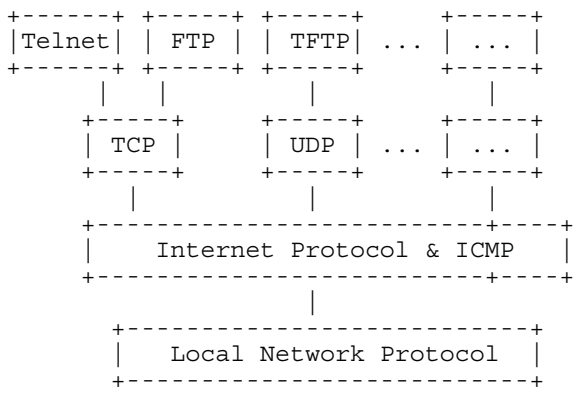
https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 031
**Keysight, EX-1045**
**Page 31**

# 2. OVERVIEW

## 2.1. Relation to Other Protocols

The following diagram illustrates the place of the internet protocol in the protocol hierarchy:




Protocol Relationships

Figure 1.

Internet protocol interfaces on one side to the higher level host-to-host protocols and on the other side to the local network protocol. In this context a "local network" may be a small network in a building or a large network such as the ARPANET.

## 2.2. Model of Operation

The model of operation for transmitting a datagram from one application program to another is illustrated by the following scenario:

We suppose that this transmission will involve one intermediate gateway.

The sending application program prepares its data and calls on its local internet module to send that data as a datagram and passes the destination address and other parameters as arguments of the call.

The internet module prepares a datagram header and attaches the data to it. The internet module determines a local network address for this internet address, in this case it is the address of a gateway.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 032
Keysight, EX-1045
Page 32

It sends this datagram and the local network address to the local network interface.

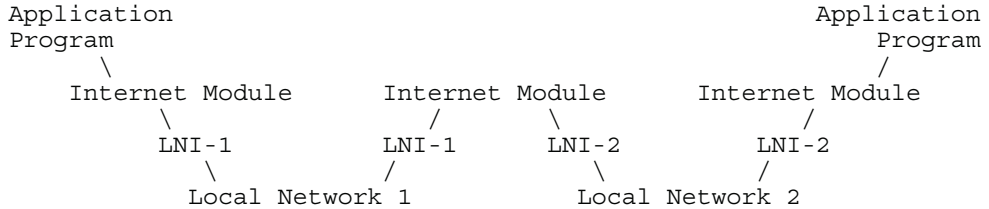
The local network interface creates a local network header, and attaches the datagram to it, then sends the result via the local network.

The datagram arrives at a gateway host wrapped in the local network header, the local network interface strips off this header, and turns the datagram over to the internet module. The internet module determines from the internet address that the datagram is to be forwarded to another host in a second network. The internet module determines a local net address for the destination host. It calls on the local network interface for that network to send the datagram.

This local network interface creates a local network header and attaches the datagram sending the result to the destination host.

At this destination host the datagram is stripped of the local net header by the local network interface and handed to the internet module.

The internet module determines that the datagram is for an application program in this host. It passes the data to the application program in response to a system call, passing the source address and other parameters as results of the call.




Transmission Path

Figure 2



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 033
Keysight, EX-1045
Page 33

### 2.3. Function Description

The function or purpose of Internet Protocol is to move datagrams through an interconnected set of networks. This is done by passing the datagrams from one internet module to another until the destination is reached. The internet modules reside in hosts and gateways in the internet system. The datagrams are routed from one internet module to another through individual networks based on the interpretation of an internet address. Thus, one important mechanism of the internet protocol is the internet address.

In the routing of messages from one internet module to another, datagrams may need to traverse a network whose maximum packet size is smaller than the size of the datagram. To overcome this difficulty, a fragmentation mechanism is provided in the internet protocol.

Addressing

A distinction is made between names, addresses, and routes [4]. A name indicates what we seek. An address indicates where it is. A route indicates how to get there. The internet protocol deals primarily with addresses. It is the task of higher level (i.e., host-to-host or application) protocols to make the mapping from names to addresses. The internet module maps internet addresses to local net addresses. It is the task of lower level (i.e., local net or gateways) procedures to make the mapping from local net addresses to routes.

Addresses are fixed length of four octets (32 bits). An address begins with a network number, followed by local address (called the "rest" field). There are three formats or classes of internet addresses: in class a, the high order bit is zero, the next 7 bits are the network, and the last 24 bits are the local address; in class b, the high order two bits are one-zero, the next 14 bits are the network and the last 16 bits are the local address; in class c, the high order three bits are one-one-zero, the next 21 bits are the network and the last 8 bits are the local address.

Care must be taken in mapping internet addresses to local net addresses; a single physical host must be able to act as if it were several distinct hosts to the extent of using several distinct internet addresses. Some hosts will also have several physical interfaces (multi-homing).

That is, provision must be made for a host to have several physical interfaces to the network with each having several logical internet addresses.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 034
Keysight, EX-1045
Page 34

Examples of address mappings may be found in "Address Mappings" [5].

## Fragmentation

Fragmentation of an internet datagram is necessary when it originates in a local net that allows a large packet size and must traverse a local net that limits packets to a smaller size to reach its destination.

An internet datagram can be marked "don't fragment." Any internet datagram so marked is not to be internet fragmented under any circumstances. If internet datagram marked don't fragment cannot be delivered to its destination without fragmenting it, it is to be discarded instead.

Fragmentation, transmission and reassembly across a local network which is invisible to the internet protocol module is called intranet fragmentation and may be used [6].

The internet fragmentation and reassembly procedure needs to be able to break a datagram into an almost arbitrary number of pieces that can be later reassembled. The receiver of the fragments uses the identification field to ensure that fragments of different datagrams are not mixed. The fragment offset field tells the receiver the position of a fragment in the original datagram. The fragment offset and length determine the portion of the original datagram covered by this fragment. The more-fragments flag indicates (by being reset) the last fragment. These fields provide sufficient information to reassemble datagrams.

The identification field is used to distinguish the fragments of one datagram from those of another. The originating protocol module of an internet datagram sets the identification field to a value that must be unique for that source-destination pair and protocol for the time the datagram will be active in the internet system. The originating protocol module of a complete datagram sets the more-fragments flag to zero and the fragment offset to zero.

To fragment a long internet datagram, an internet protocol module (for example, in a gateway), creates two new internet datagrams and copies the contents of the internet header fields from the long datagram into both new internet headers. The data of the long datagram is divided into two portions on a 8 octet (64 bit) boundary (the second portion might not be an integral multiple of 8 octets, but the first must be). Call the number of 8 octet blocks in the first portion NFB (for Number of Fragment Blocks). The first portion of the data is placed in the first new internet datagram, and the total length field is set to the length of the first



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 035
Keysight, EX-1045
Page 35

datagram. The more-fragments flag is set to one. The second portion of the data is placed in the second new internet datagram, and the total length field is set to the length of the second datagram. The more-fragments flag carries the same value as the long datagram. The fragment offset field of the second new internet datagram is set to the value of that field in the long datagram plus NFB.

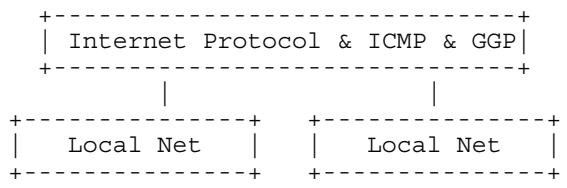
This procedure can be generalized for an n-way split, rather than the two-way split described.

To assemble the fragments of an internet datagram, an internet protocol module (for example at a destination host) combines internet datagrams that all have the same value for the four fields: identification, source, destination, and protocol. The combination is done by placing the data portion of each fragment in the relative position indicated by the fragment offset in that fragment's internet header. The first fragment will have the fragment offset zero, and the last fragment will have the more-fragments flag reset to zero.

### 2.4. Gateways

Gateways implement internet protocol to forward datagrams between networks. Gateways also implement the Gateway to Gateway Protocol (GGP) [7] to coordinate routing and other internet control information.

In a gateway the higher level protocols need not be implemented and the GGP functions are added to the IP module.




Gateway Protocols

Figure 3.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 036
Keysight, EX-1045
Page 36

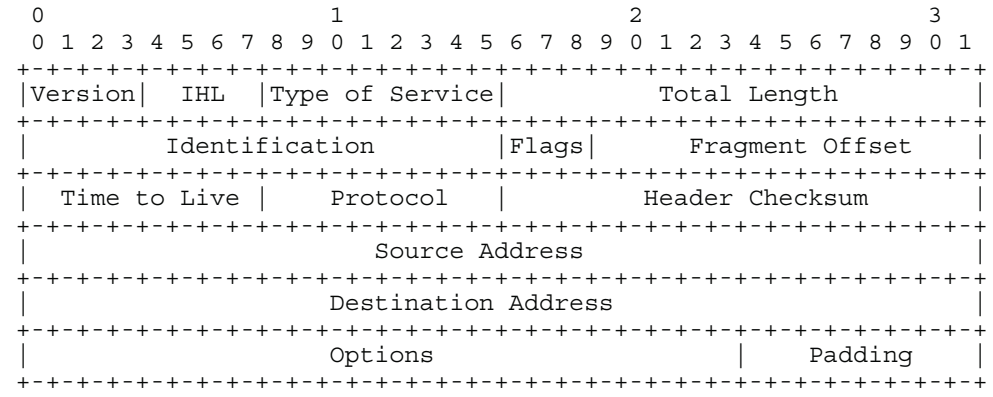
https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 037
Keysight, EX-1045
Page 37

## 3. SPECIFICATION

### 3.1. Internet Header Format

A summary of the contents of the internet header follows:




Example Internet Datagram Header

Figure 4.

Note that each tick mark represents one bit position.

Version: 4 bits

The Version field indicates the format of the internet header. This document describes version 4.

IHL: 4 bits

Internet Header Length is the length of the internet header in 32 bit words, and thus points to the beginning of the data. Note that the minimum value for a correct header is 5.



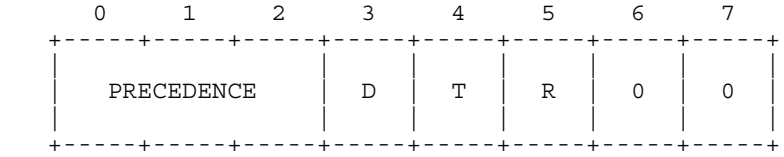
https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 038
Keysight, EX-1045
Page 38

Type of Service: 8 bits

The Type of Service provides an indication of the abstract parameters of the quality of service desired. These parameters are to be used to guide the selection of the actual service parameters when transmitting a datagram through a particular network. Several networks offer service precedence, which somehow treats high precedence traffic as more important than other traffic (generally by accepting only traffic above a certain precedence at time of high load). The major choice is a three way tradeoff between low-delay, high-reliability, and high-throughput.

```
Bits 0-2:  Precedence.
Bit    3:  0 = Normal Delay,      1 = Low Delay.
Bits   4:  0 = Normal Throughput, 1 = High Throughput.
Bits   5:  0 = Normal Relibility, 1 = High Relibility.
Bit  6-7:  Reserved for Future Use.
```




Precedence

```
111 - Network Control
110 - Internetwork Control
101 - CRITIC/ECP
100 - Flash Override
011 - Flash
010 - Immediate
001 - Priority
000 - Routine
```

The use of the Delay, Throughput, and Reliability indications may increase the cost (in some sense) of the service. In many networks better performance for one of these parameters is coupled with worse performance on another. Except for very unusual cases at most two of these three indications should be set.

The type of service is used to specify the treatment of the datagram during its transmission through the internet system. Example mappings of the internet type of service to the actual service provided on networks such as AUTODIN II, ARPANET, SATNET, and PRNET is given in "Service Mappings" [8].



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 039
Keysight, EX-1045
Page 39

The Network Control precedence designation is intended to be used within a network only. The actual use and control of that designation is up to each network. The Internetwork Control designation is intended for use by gateway control originators only. If the actual use of these precedence designations is of concern to a particular network, it is the responsibility of that network to control the access to, and use of, those precedence designations.

Total Length: 16 bits

Total Length is the length of the datagram, measured in octets, including internet header and data. This field allows the length of a datagram to be up to 65,535 octets. Such long datagrams are impractical for most hosts and networks. All hosts must be prepared to accept datagrams of up to 576 octets (whether they arrive whole or in fragments). It is recommended that hosts only send datagrams larger than 576 octets if they have assurance that the destination is prepared to accept the larger datagrams.

The number 576 is selected to allow a reasonable sized data block to be transmitted in addition to the required header information. For example, this size allows a data block of 512 octets plus 64 header octets to fit in a datagram. The maximal internet header is 60 octets, and a typical internet header is 20 octets, allowing a margin for headers of higher level protocols.

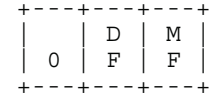
Identification: 16 bits

An identifying value assigned by the sender to aid in assembling the fragments of a datagram.

Flags: 3 bits

Various Control Flags.

Bit 0: reserved, must be zero
Bit 1: (DF) 0 = May Fragment, 1 = Don't Fragment.
Bit 2: (MF) 0 = Last Fragment, 1 = More Fragments.




Fragment Offset: 13 bits

This field indicates where in the datagram this fragment belongs.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 040
Keysight, EX-1045
Page 40

The fragment offset is measured in units of 8 octets (64 bits). The first fragment has offset zero.

Time to Live: 8 bits

This field indicates the maximum time the datagram is allowed to remain in the internet system. If this field contains the value zero, then the datagram must be destroyed. This field is modified in internet header processing. The time is measured in units of seconds, but since every module that processes a datagram must decrease the TTL by at least one even if it process the datagram in less than a second, the TTL must be thought of only as an upper bound on the time a datagram may exist. The intention is to cause undeliverable datagrams to be discarded, and to bound the maximum datagram lifetime.

Protocol: 8 bits

This field indicates the next level protocol used in the data portion of the internet datagram. The values for various protocols are specified in "Assigned Numbers" [9].

Header Checksum: 16 bits

A checksum on the header only. Since some header fields change (e.g., time to live), this is recomputed and verified at each point that the internet header is processed.

The checksum algorithm is:

The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header. For purposes of computing the checksum, the value of the checksum field is zero.

This is a simple to compute checksum and experimental evidence indicates it is adequate, but it is provisional and may be replaced by a CRC procedure, depending on further experience.

Source Address: 32 bits

The source address. See section 3.2.

Destination Address: 32 bits

The destination address. See section 3.2.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 041
Keysight, EX-1045
Page 41

Options: variable

The options may appear or not in datagrams. They must be implemented by all IP modules (host and gateways). What is optional is their transmission in any particular datagram, not their implementation.

In some environments the security option may be required in all datagrams.

The option field is variable in length. There may be zero or more options. There are two cases for the format of an option:

Case 1: A single octet of option-type.

Case 2: An option-type octet, an option-length octet, and the actual option-data octets.

The option-length octet counts the option-type octet and the option-length octet as well as the option-data octets.

The option-type octet is viewed as having 3 fields:

1 bit copied flag,
2 bits option class,
5 bits option number.

The copied flag indicates that this option is copied into all fragments on fragmentation.

0 = not copied
1 = copied

The option classes are:

0 = control
1 = reserved for future use
2 = debugging and measurement
3 = reserved for future use



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 042
Keysight, EX-1045
Page 42

The following internet options are defined:

| CLASS | NUMBER | LENGTH | DESCRIPTION |
|---|---|---|---|
| 0 | 0 | - | End of Option list. This option occupies only 1 octet; it has no length octet. |
| 0 | 1 | - | No Operation. This option occupies only 1 octet; it has no length octet. |
| 0 | 2 | 11 | Security. Used to carry Security, Compartmentation, User Group (TCC), and Handling Restriction Codes compatible with DOD requirements. |
| 0 | 3 | var. | Loose Source Routing. Used to route the internet datagram based on information supplied by the source. |
| 0 | 9 | var. | Strict Source Routing. Used to route the internet datagram based on information supplied by the source. |
| 0 | 7 | var. | Record Route. Used to trace the route an internet datagram takes. |
| 0 | 8 | 4 | Stream ID. Used to carry the stream identifier. |
| 2 | 4 | var. | Internet Timestamp. |

Specific Option Definitions

End of Option List




This option indicates the end of the option list. This might not coincide with the end of the internet header according to the internet header length. This is used at the end of all options, not the end of each option, and need only be used if the end of the options would not otherwise coincide with the end of the internet header.

May be copied, introduced, or deleted on fragmentation, or for any other reason.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 043
Keysight, EX-1045
Page 43

No Operation

```
+--------+
|00000001|
+--------+
  Type=1
```

This option may be used between options, for example, to align the beginning of a subsequent option on a 32 bit boundary.

May be copied, introduced, or deleted on fragmentation, or for any other reason.

Security

This option provides a way for hosts to send security, compartmentation, handling restrictions, and TCC (closed user group) parameters. The format for this option is as follows:

```
+--------+--------+---//---+---//---+---//---+---//---+
|10000010|00001011|SSS  SSS|CCC  CCC|HHH  HHH|  TCC   |
+--------+--------+---//---+---//---+---//---+---//---+
 Type=130 Length=11
```

Security (S field): 16 bits

Specifies one of 16 levels of security (eight of which are reserved for future use).

```
00000000 00000000 - Unclassified
11110001 00110101 - Confidential
01111000 10011010 - EFTO
10111100 01001101 - MMMM
01011110 00100110 - PROG
10101111 00010011 - Restricted
11010111 10001000 - Secret
01101011 11000101 - Top Secret
00110101 11100010 - (Reserved for future use)
10011010 11110001 - (Reserved for future use)
01001101 01111000 - (Reserved for future use)
00100100 10111101 - (Reserved for future use)
00010011 01011110 - (Reserved for future use)
10001001 10101111 - (Reserved for future use)
11000100 11010110 - (Reserved for future use)
11100010 01101011 - (Reserved for future use)
```



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 044
Keysight, EX-1045
Page 44

Compartments (C field): 16 bits

An all zero value is used when the information transmitted is not compartmented. Other values for the compartments field may be obtained from the Defense Intelligence Agency.

Handling Restrictions (H field): 16 bits

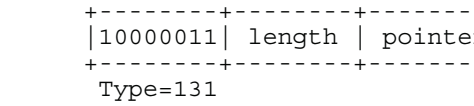
The values for the control and release markings are alphanumeric digraphs and are defined in the Defense Intelligence Agency Manual DIAM 65-19, "Standard Security Markings".

Transmission Control Code (TCC field): 24 bits

Provides a means to segregate traffic and define controlled communities of interest among subscribers. The TCC values are trigraphs, and are available from HQ DCA Code 530.

Must be copied on fragmentation. This option appears at most once in a datagram.

Loose Source and Record Route




The loose source and record route (LSRR) option provides a means for the source of an internet datagram to supply routing information to be used by the gateways in forwarding the datagram to the destination, and to record the route information.

The option begins with the option type code. The second octet is the option length which includes the option type code and the length octet, the pointer octet, and length-3 octets of route data. The third octet is the pointer into the route data indicating the octet which begins the next source address to be processed. The pointer is relative to this option, and the smallest legal value for the pointer is 4.

A route data is composed of a series of internet addresses. Each internet address is 32 bits or 4 octets. If the pointer is greater than the length, the source route is empty (and the recorded route full) and the routing is to be based on the destination address field.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 045
Keysight, EX-1045
Page 45

If the address in destination address field has been reached and the pointer is not greater than the length, the next address in the source route replaces the address in the destination address field, and the recorded route address replaces the source address just used, and pointer is increased by four.

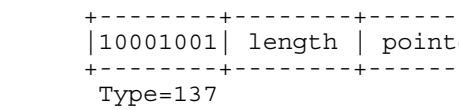
The recorded route address is the internet module's own internet address as known in the environment into which this datagram is being forwarded.

This procedure of replacing the source route with the recorded route (though it is in the reverse of the order it must be in to be used as a source route) means the option (and the IP header as a whole) remains a constant length as the datagram progresses through the internet.

This option is a loose source route because the gateway or host IP is allowed to use any route of any number of other intermediate gateways to reach the next address in the route.

Must be copied on fragmentation. Appears at most once in a datagram.

Strict Source and Record Route




The strict source and record route (SSRR) option provides a means for the source of an internet datagram to supply routing information to be used by the gateways in forwarding the datagram to the destination, and to record the route information.

The option begins with the option type code. The second octet is the option length which includes the option type code and the length octet, the pointer octet, and length-3 octets of route data. The third octet is the pointer into the route data indicating the octet which begins the next source address to be processed. The pointer is relative to this option, and the smallest legal value for the pointer is 4.

A route data is composed of a series of internet addresses. Each internet address is 32 bits or 4 octets. If the pointer is greater than the length, the source route is empty (and the



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 046
Keysight, EX-1045
Page 46

recorded route full) and the routing is to be based on the destination address field.

If the address in destination address field has been reached and the pointer is not greater than the length, the next address in the source route replaces the address in the destination address field, and the recorded route address replaces the source address just used, and pointer is increased by four.

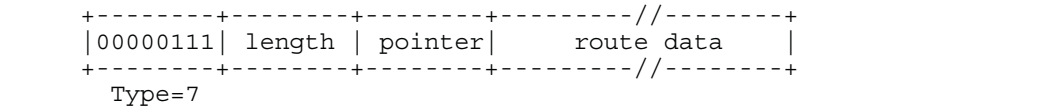
The recorded route address is the internet module's own internet address as known in the environment into which this datagram is being forwarded.

This procedure of replacing the source route with the recorded route (though it is in the reverse of the order it must be in to be used as a source route) means the option (and the IP header as a whole) remains a constant length as the datagram progresses through the internet.

This option is a strict source route because the gateway or host IP must send the datagram directly to the next address in the source route through only the directly connected network indicated in the next address to reach the next gateway or host specified in the route.

Must be copied on fragmentation. Appears at most once in a datagram.

Record Route




The record route option provides a means to record the route of an internet datagram.

The option begins with the option type code. The second octet is the option length which includes the option type code and the length octet, the pointer octet, and length-3 octets of route data. The third octet is the pointer into the route data indicating the octet which begins the next area to store a route address. The pointer is relative to this option, and the smallest legal value for the pointer is 4.

A recorded route is composed of a series of internet addresses. Each internet address is 32 bits or 4 octets. If the pointer is



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 047
Keysight, EX-1045
Page 47

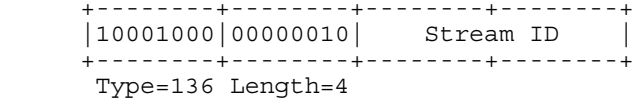
greater than the length, the recorded route data area is full. The originating host must compose this option with a large enough route data area to hold all the address expected. The size of the option does not change due to adding addresses. The intitial contents of the route data area must be zero.

When an internet module routes a datagram it checks to see if the record route option is present. If it is, it inserts its own internet address as known in the environment into which this datagram is being forwarded into the recorded route begining at the octet indicated by the pointer, and increments the pointer by four.

If the route data area is already full (the pointer exceeds the length) the datagram is forwarded without inserting the address into the recorded route. If there is some room but not enough room for a full address to be inserted, the original datagram is considered to be in error and is discarded. In either case an ICMP parameter problem message may be sent to the source host [3].

Not copied on fragmentation, goes in first fragment only. Appears at most once in a datagram.

Stream Identifier




This option provides a way for the 16-bit SATNET stream identifier to be carried through networks that do not support the stream concept.

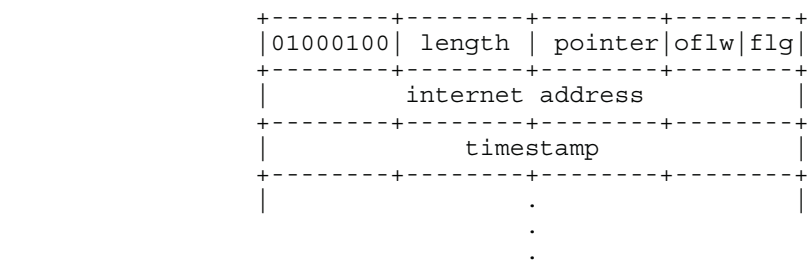
Must be copied on fragmentation. Appears at most once in a datagram.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 048
Keysight, EX-1045
Page 48

Internet Timestamp




Type = 68

The Option Length is the number of octets in the option counting the type, length, pointer, and overflow/flag octets (maximum length 40).

The Pointer is the number of octets from the beginning of this option to the end of timestamps plus one (i.e., it points to the octet beginning the space for next timestamp). The smallest legal value is 5. The timestamp area is full when the pointer is greater than the length.

The Overflow (oflw) [4 bits] is the number of IP modules that cannot register timestamps due to lack of space.

The Flag (flg) [4 bits] values are

0 -- time stamps only, stored in consecutive 32-bit words,

1 -- each timestamp is preceded with internet address of the registering entity,

3 -- the internet address fields are prespecified. An IP module only registers its timestamp if it matches its own address with the next specified internet address.

The Timestamp is a right-justified, 32-bit timestamp in milliseconds since midnight UT. If the time is not available in milliseconds or cannot be provided with respect to midnight UT then any time may be inserted as a timestamp provided the high order bit of the timestamp field is set to one to indicate the use of a non-standard value.

The originating host must compose this option with a large enough timestamp data area to hold all the timestamp information expected. The size of the option does not change due to adding



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 049
Keysight, EX-1045
Page 49

timestamps. The intitial contents of the timestamp data area must be zero or internet address/zero pairs.

If the timestamp data area is already full (the pointer exceeds the length) the datagram is forwarded without inserting the timestamp, but the overflow count is incremented by one.

If there is some room but not enough room for a full timestamp to be inserted, or the overflow count itself overflows, the original datagram is considered to be in error and is discarded. In either case an ICMP parameter problem message may be sent to the source host [3].

The timestamp option is not copied upon fragmentation. It is carried in the first fragment. Appears at most once in a datagram.

Padding: variable

The internet header padding is used to ensure that the internet header ends on a 32 bit boundary. The padding is zero.

### 3.2. Discussion

The implementation of a protocol must be robust. Each implementation must expect to interoperate with others created by different individuals. While the goal of this specification is to be explicit about the protocol there is the possibility of differing interpretations. In general, an implementation must be conservative in its sending behavior, and liberal in its receiving behavior. That is, it must be careful to send well-formed datagrams, but must accept any datagram that it can interpret (e.g., not object to technical errors where the meaning is still clear).

The basic internet service is datagram oriented and provides for the fragmentation of datagrams at gateways, with reassembly taking place at the destination internet protocol module in the destination host. Of course, fragmentation and reassembly of datagrams within a network or by private agreement between the gateways of a network is also allowed since this is transparent to the internet protocols and the higher-level protocols. This transparent type of fragmentation and reassembly is termed "network-dependent" (or intranet) fragmentation and is not discussed further here.

Internet addresses distinguish sources and destinations to the host level and provide a protocol field as well. It is assumed that each protocol will provide for whatever multiplexing is necessary within a host.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 050
Keysight, EX-1045
Page 50

Addressing

To provide for flexibility in assigning address to networks and allow for the large number of small to intermediate sized networks the interpretation of the address field is coded to specify a small number of networks with a large number of host, a moderate number of networks with a moderate number of hosts, and a large number of networks with a small number of hosts. In addition there is an escape code for extended addressing mode.

Address Formats:

| High Order Bits | Format | Class |
|---|---|---|
| 0 | 7 bits of net, 24 bits of host | a |
| 10 | 14 bits of net, 16 bits of host | b |
| 110 | 21 bits of net, 8 bits of host | c |
| 111 | escape to extended addressing mode | |

A value of zero in the network field means this network. This is only used in certain ICMP messages. The extended addressing mode is undefined. Both of these features are reserved for future use.

The actual values assigned for network addresses is given in "Assigned Numbers" [9].

The local address, assigned by the local network, must allow for a single physical host to act as several distinct internet hosts. That is, there must be a mapping between internet host addresses and network/host interfaces that allows several internet addresses to correspond to one interface. It must also be allowed for a host to have several physical interfaces and to treat the datagrams from several of them as if they were all addressed to a single host.

Address mappings between internet addresses and addresses for ARPANET, SATNET, PRNET, and other networks are described in "Address Mappings" [5].

Fragmentation and Reassembly.

The internet identification field (ID) is used together with the source and destination address, and the protocol fields, to identify datagram fragments for reassembly.

The More Fragments flag bit (MF) is set if the datagram is not the last fragment. The Fragment Offset field identifies the fragment location, relative to the beginning of the original unfragmented datagram. Fragments are counted in units of 8 octets. The



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 051
Keysight, EX-1045
Page 51

fragmentation strategy is designed so than an unfragmented datagram has all zero fragmentation information (MF = 0, fragment offset = 0). If an internet datagram is fragmented, its data portion must be broken on 8 octet boundaries.

This format allows 2**13 = 8192 fragments of 8 octets each for a total of 65,536 octets. Note that this is consistent with the the datagram total length field (of course, the header is counted in the total length and not in the fragments).

When fragmentation occurs, some options are copied, but others remain with the first fragment only.

Every internet module must be able to forward a datagram of 68 octets without further fragmentation. This is because an internet header may be up to 60 octets, and the minimum fragment is 8 octets.

Every internet destination must be able to receive a datagram of 576 octets either in one piece or in fragments to be reassembled.

The fields which may be affected by fragmentation include:

(1) options field
(2) more fragments flag
(3) fragment offset
(4) internet header length field
(5) total length field
(6) header checksum

If the Don't Fragment flag (DF) bit is set, then internet fragmentation of this datagram is NOT permitted, although it may be discarded. This can be used to prohibit fragmentation in cases where the receiving host does not have sufficient resources to reassemble internet fragments.

One example of use of the Don't Fragment feature is to down line load a small host. A small host could have a boot strap program that accepts a datagram stores it in memory and then executes it.

The fragmentation and reassembly procedures are most easily described by examples. The following procedures are example implementations.

General notation in the following pseudo programs: "=<" means "less than or equal", "#" means "not equal", "=" means "equal", "<-" means "is set to". Also, "x to y" includes x and excludes y; for example, "4 to 7" would include 4, 5, and 6 (but not 7).



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 052
Keysight, EX-1045
Page 52

An Example Fragmentation Procedure

The maximum sized datagram that can be transmitted through the next network is called the maximum transmission unit (MTU).

If the total length is less than or equal the maximum transmission unit then submit this datagram to the next step in datagram processing; otherwise cut the datagram into two fragments, the first fragment being the maximum size, and the second fragment being the rest of the datagram. The first fragment is submitted to the next step in datagram processing, while the second fragment is submitted to this procedure in case it is still too large.

Notation:

```
FO    -  Fragment Offset
IHL   -  Internet Header Length
DF    -  Don't Fragment flag
MF    -  More Fragments flag
TL    -  Total Length
OFO   -  Old Fragment Offset
OIHL  -  Old Internet Header Length
OMF   -  Old More Fragments flag
OTL   -  Old Total Length
NFB   -  Number of Fragment Blocks
MTU   -  Maximum Transmission Unit
```

Procedure:

```
IF TL =< MTU THEN Submit this datagram to the next step
     in datagram processing ELSE IF DF = 1 THEN discard the
datagram ELSE
To produce the first fragment:
(1)  Copy the original internet header;
(2)  OIHL <- IHL; OTL <- TL; OFO <- FO; OMF <- MF;
(3)  NFB <- (MTU-IHL*4)/8;
(4)  Attach the first NFB*8 data octets;
(5)  Correct the header:
     MF <- 1;  TL <- (IHL*4)+(NFB*8);
     Recompute Checksum;
(6)  Submit this fragment to the next step in
     datagram processing;
To produce the second fragment:
(7)  Selectively copy the internet header (some options
     are not copied, see option definitions);
(8)  Append the remaining data;
(9)  Correct the header:
     IHL <- (((OIHL*4)-(length of options not copied))+3)/4;
```



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 053
Keysight, EX-1045
Page 53

```
        TL <- OTL - NFB*8 - (OIHL-IHL)*4);
        FO <- OFO + NFB;  MF <- OMF;  Recompute Checksum;
   (10) Submit this fragment to the fragmentation test; DONE.
```

In the above procedure each fragment (except the last) was made the maximum allowable size. An alternative might produce less than the maximum size datagrams. For example, one could implement a fragmentation procedure that repeatly divided large datagrams in half until the resulting fragments were less than the maximum transmission unit size.

An Example Reassembly Procedure

For each datagram the buffer identifier is computed as the concatenation of the source, destination, protocol, and identification fields. If this is a whole datagram (that is both the fragment offset and the more fragments fields are zero), then any reassembly resources associated with this buffer identifier are released and the datagram is forwarded to the next step in datagram processing.

If no other fragment with this buffer identifier is on hand then reassembly resources are allocated. The reassembly resources consist of a data buffer, a header buffer, a fragment block bit table, a total data length field, and a timer. The data from the fragment is placed in the data buffer according to its fragment offset and length, and bits are set in the fragment block bit table corresponding to the fragment blocks received.

If this is the first fragment (that is the fragment offset is zero) this header is placed in the header buffer. If this is the last fragment ( that is the more fragments field is zero) the total data length is computed. If this fragment completes the datagram (tested by checking the bits set in the fragment block table), then the datagram is sent to the next step in datagram processing; otherwise the timer is set to the maximum of the current timer value and the value of the time to live field from this fragment; and the reassembly routine gives up control.

If the timer runs out, the all reassembly resources for this buffer identifier are released. The initial setting of the timer is a lower bound on the reassembly waiting time. This is because the waiting time will be increased if the Time to Live in the arriving fragment is greater than the current timer value but will not be decreased if it is less. The maximum this timer value could reach is the maximum time to live (approximately 4.25 minutes). The current recommendation for the initial timer setting is 15 seconds. This may be changed as experience with



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 054
Keysight, EX-1045
Page 54

this protocol accumulates. Note that the choice of this parameter value is related to the buffer capacity available and the data rate of the transmission medium; that is, data rate times timer value equals buffer size (e.g., 10Kb/s X 15s = 150Kb).

Notation:

```
  FO    -  Fragment Offset
  IHL   -  Internet Header Length
  MF    -  More Fragments flag
  TTL   -  Time To Live
  NFB   -  Number of Fragment Blocks
  TL    -  Total Length
  TDL   -  Total Data Length
  BUFID -  Buffer Identifier
  RCVBT -  Fragment Received Bit Table
  TLB   -  Timer Lower Bound
```

Procedure:

```
  (1)  BUFID <- source|destination|protocol|identification;
  (2)  IF FO = 0 AND MF = 0
  (3)     THEN IF buffer with BUFID is allocated
  (4)             THEN flush all reassembly for this BUFID;
  (5)          Submit datagram to next step; DONE.
  (6)     ELSE IF no buffer with BUFID is allocated
  (7)             THEN allocate reassembly resources
                       with BUFID;
                       TIMER <- TLB; TDL <- 0;
  (8)          put data from fragment into data buffer with
               BUFID from octet FO*8 to
                                 octet (TL-(IHL*4))+FO*8;
  (9)          set RCVBT bits from FO
                                to FO+((TL-(IHL*4)+7)/8);
  (10)         IF MF = 0 THEN TDL <- TL-(IHL*4)+(FO*8)
  (11)         IF FO = 0 THEN put header in header buffer
  (12)         IF TDL # 0
  (13)          AND all RCVBT bits from 0
                                    to (TDL+7)/8 are set
  (14)            THEN TL <- TDL+(IHL*4)
  (15)                 Submit datagram to next step;
  (16)                 free all reassembly resources
                       for this BUFID; DONE.
  (17)         TIMER <- MAX(TIMER,TTL);
  (18)         give up until next fragment or timer expires;
  (19) timer expires: flush all reassembly with this BUFID; DONE.
```

In the case that two or more fragments contain the same data



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 055
Keysight, EX-1045
Page 55

either identically or through a partial overlap, this procedure will use the more recently arrived copy in the data buffer and datagram delivered.

Identification

The choice of the Identifier for a datagram is based on the need to provide a way to uniquely identify the fragments of a particular datagram. The protocol module assembling fragments judges fragments to belong to the same datagram if they have the same source, destination, protocol, and Identifier. Thus, the sender must choose the Identifier to be unique for this source, destination pair and protocol for the time the datagram (or any fragment of it) could be alive in the internet.

It seems then that a sending protocol module needs to keep a table of Identifiers, one entry for each destination it has communicated with in the last maximum packet lifetime for the internet.

However, since the Identifier field allows 65,536 different values, some host may be able to simply use unique identifiers independent of destination.

It is appropriate for some higher level protocols to choose the identifier. For example, TCP protocol modules may retransmit an identical TCP segment, and the probability for correct reception would be enhanced if the retransmission carried the same identifier as the original transmission since fragments of either datagram could be used to construct a correct TCP segment.

Type of Service

The type of service (TOS) is for internet service quality selection. The type of service is specified along the abstract parameters precedence, delay, throughput, and reliability. These abstract parameters are to be mapped into the actual service parameters of the particular networks the datagram traverses.

Precedence. An independent measure of the importance of this datagram.

Delay. Prompt delivery is important for datagrams with this indication.

Throughput. High data rate is important for datagrams with this indication.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 056
Keysight, EX-1045
Page 56

Reliability. A higher level of effort to ensure delivery is important for datagrams with this indication.

For example, the ARPANET has a priority bit, and a choice between "standard" messages (type 0) and "uncontrolled" messages (type 3), (the choice between single packet and multipacket messages can also be considered a service parameter). The uncontrolled messages tend to be less reliably delivered and suffer less delay. Suppose an internet datagram is to be sent through the ARPANET. Let the internet type of service be given as:

| | |
|---|---|
| Precedence: | 5 |
| Delay: | 0 |
| Throughput: | 1 |
| Reliability: | 1 |

In this example, the mapping of these parameters to those available for the ARPANET would be to set the ARPANET priority bit on since the Internet precedence is in the upper half of its range, to select standard messages since the throughput and reliability requirements are indicated and delay is not. More details are given on service mappings in "Service Mappings" [8].

Time to Live

The time to live is set by the sender to the maximum time the datagram is allowed to be in the internet system. If the datagram is in the internet system longer than the time to live, then the datagram must be destroyed.

This field must be decreased at each point that the internet header is processed to reflect the time spent processing the datagram. Even if no local information is available on the time actually spent, the field must be decremented by 1. The time is measured in units of seconds (i.e. the value 1 means one second). Thus, the maximum time to live is 255 seconds or 4.25 minutes. Since every module that processes a datagram must decrease the TTL by at least one even if it process the datagram in less than a second, the TTL must be thought of only as an upper bound on the time a datagram may exist. The intention is to cause undeliverable datagrams to be discarded, and to bound the maximum datagram lifetime.

Some higher level reliable connection protocols are based on assumptions that old duplicate datagrams will not arrive after a certain time elapses. The TTL is a way for such protocols to have an assurance that their assumption is met.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 057
Keysight, EX-1045
Page 57

Options

The options are optional in each datagram, but required in implementations. That is, the presence or absence of an option is the choice of the sender, but each internet module must be able to parse every option. There can be several options present in the option field.

The options might not end on a 32-bit boundary. The internet header must be filled out with octets of zeros. The first of these would be interpreted as the end-of-options option, and the remainder as internet header padding.

Every internet module must be able to act on every option. The Security Option is required if classified, restricted, or compartmented traffic is to be passed.

Checksum

The internet header checksum is recomputed if the internet header is changed. For example, a reduction of the time to live, additions or changes to internet options, or due to fragmentation. This checksum at the internet level is intended to protect the internet header fields from transmission errors.

There are some applications where a few data bit errors are acceptable while retransmission delays are not. If the internet protocol enforced data correctness such applications could not be supported.

Errors

Internet protocol errors may be reported via the ICMP messages [3].

### 3.3. Interfaces

The functional description of user interfaces to the IP is, at best, fictional, since every operating system will have different facilities. Consequently, we must warn readers that different IP implementations may have different user interfaces. However, all IPs must provide a certain minimum set of services to guarantee that all IP implementations can support the same protocol hierarchy. This section specifies the functional interfaces required of all IP implementations.

Internet protocol interfaces on one side to the local network and on the other side to either a higher level protocol or an application program. In the following, the higher level protocol or application



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 058
Keysight, EX-1045
Page 58

program (or even a gateway program) will be called the "user" since it is using the internet module. Since internet protocol is a datagram protocol, there is minimal memory or state maintained between datagram transmissions, and each call on the internet protocol module by the user supplies all information necessary for the IP to perform the service requested.

An Example Upper Level Interface

The following two example calls satisfy the requirements for the user to internet protocol module communication ("=>" means returns):

```
SEND (src, dst, prot, TOS, TTL, BufPTR, len, Id, DF, opt => result)

  where:

    src = source address
    dst = destination address
    prot = protocol
    TOS = type of service
    TTL = time to live
    BufPTR = buffer pointer
    len = length of buffer
    Id  = Identifier
    DF = Don't Fragment
    opt = option data
    result = response
      OK = datagram sent ok
      Error = error in arguments or local network error

  Note that the precedence is included in the TOS and the
  security/compartment is passed as an option.

RECV (BufPTR, prot, => result, src, dst, TOS, len, opt)

  where:

    BufPTR = buffer pointer
    prot = protocol
    result = response
      OK = datagram received ok
      Error = error in arguments
    len = length of buffer
    src = source address
    dst = destination address
    TOS = type of service
    opt = option data
```



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 059
Keysight, EX-1045
Page 59

When the user sends a datagram, it executes the SEND call supplying all the arguments. The internet protocol module, on receiving this call, checks the arguments and prepares and sends the message. If the arguments are good and the datagram is accepted by the local network, the call returns successfully. If either the arguments are bad, or the datagram is not accepted by the local network, the call returns unsuccessfully. On unsuccessful returns, a reasonable report must be made as to the cause of the problem, but the details of such reports are up to individual implementations.

When a datagram arrives at the internet protocol module from the local network, either there is a pending RECV call from the user addressed or there is not. In the first case, the pending call is satisfied by passing the information from the datagram to the user. In the second case, the user addressed is notified of a pending datagram. If the user addressed does not exist, an ICMP error message is returned to the sender, and the data is discarded.

The notification of a user may be via a pseudo interrupt or similar mechanism, as appropriate in the particular operating system environment of the implementation.

A user's RECV call may then either be immediately satisfied by a pending datagram, or the call may be pending until a datagram arrives.

The source address is included in the send call in case the sending host has several addresses (multiple physical connections or logical addresses). The internet module must check to see that the source address is one of the legal address for this host.

An implementation may also allow or require a call to the internet module to indicate interest in or reserve exclusive use of a class of datagrams (e.g., all those with a certain value in the protocol field).

This section functionally characterizes a USER/IP interface. The notation used is similar to most procedure of function calls in high level languages, but this usage is not meant to rule out trap type service calls (e.g., SVCs, UUOs, EMTs), or any other form of interprocess communication.



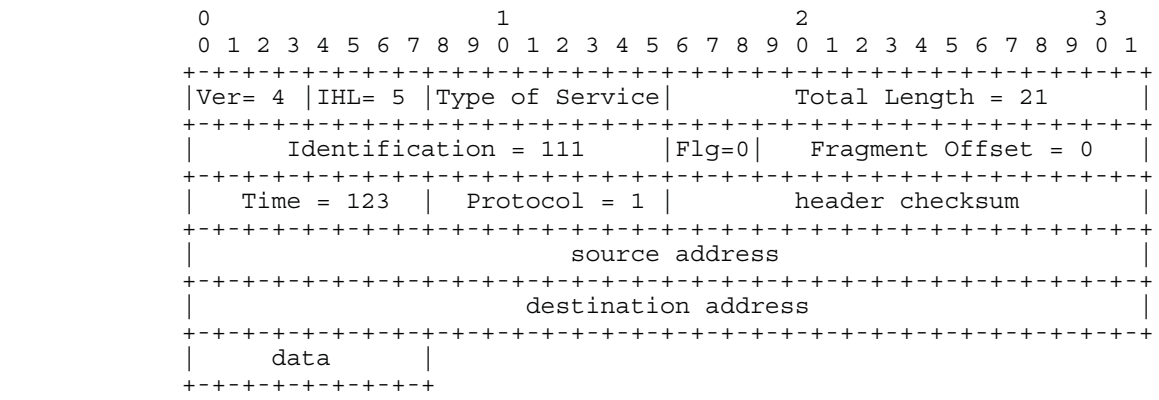
https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 060
**Keysight, EX-1045**
**Page 60**

APPENDIX A: Examples & Scenarios

Example 1:

This is an example of the minimal data carrying internet datagram:




Example Internet Datagram

Figure 5.

Note that each tick mark represents one bit position.

This is a internet datagram in version 4 of internet protocol; the internet header consists of five 32 bit words, and the total length of the datagram is 21 octets. This datagram is a complete datagram (not a fragment).

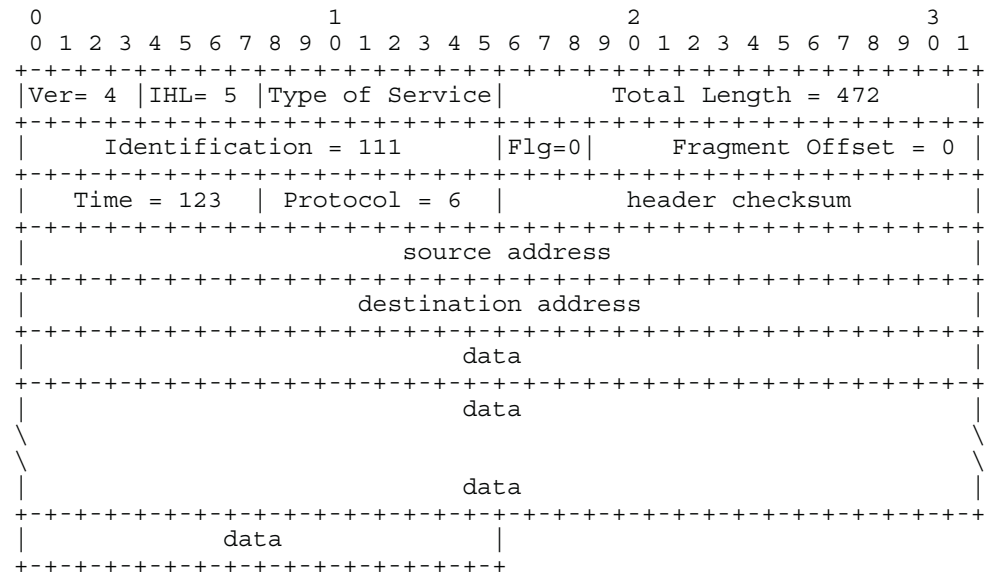


https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 061
Keysight, EX-1045
Page 61

Example 2:

In this example, we show first a moderate size internet datagram (452 data octets), then two internet fragments that might result from the fragmentation of this datagram if the maximum sized transmission allowed were 280 octets.




Example Internet Datagram

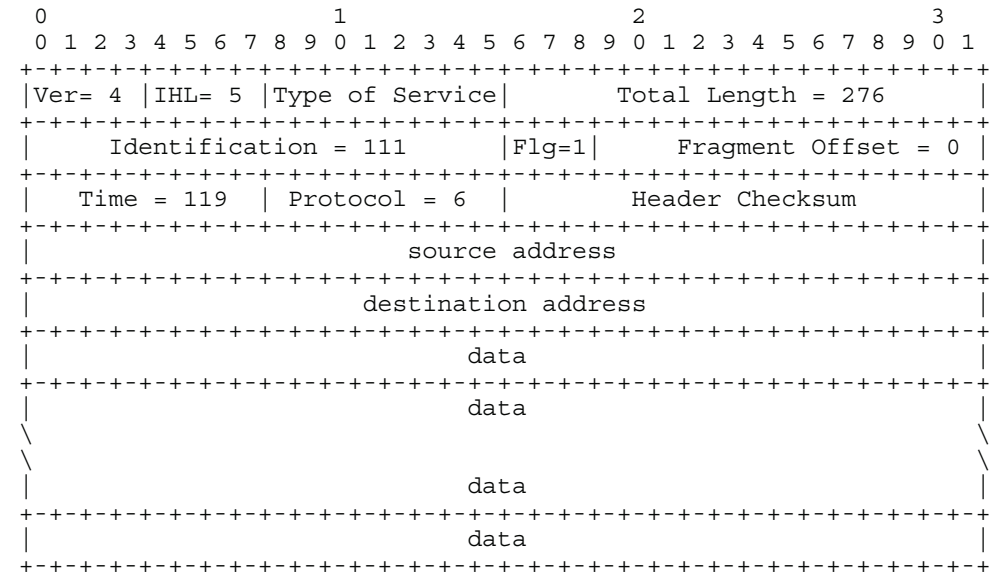
Figure 6.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 062
Keysight, EX-1045
Page 62

Now the first fragment that results from splitting the datagram after 256 data octets.




Example Internet Fragment

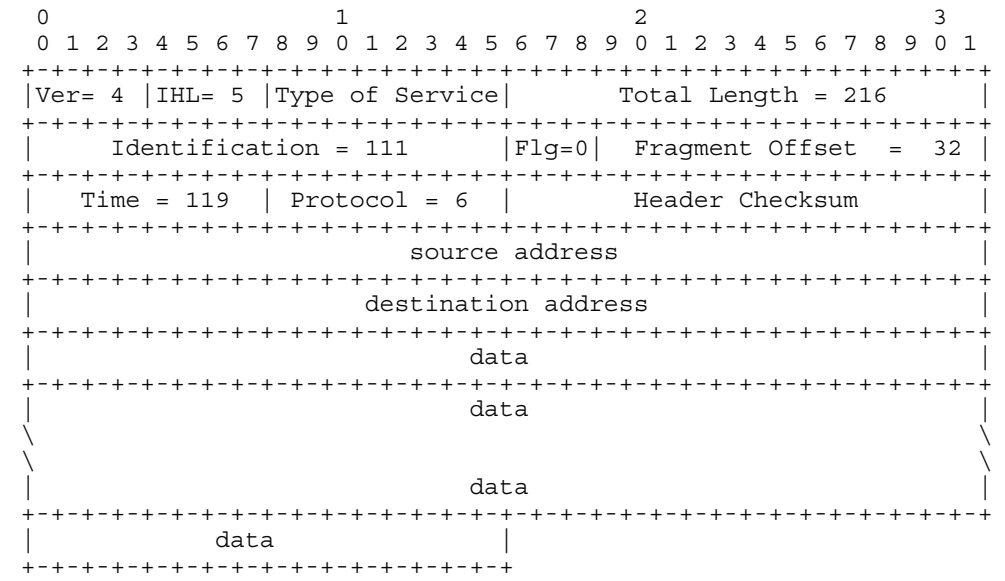
Figure 7.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 063
Keysight, EX-1045
Page 63

And the second fragment.




Example Internet Fragment

Figure 8.

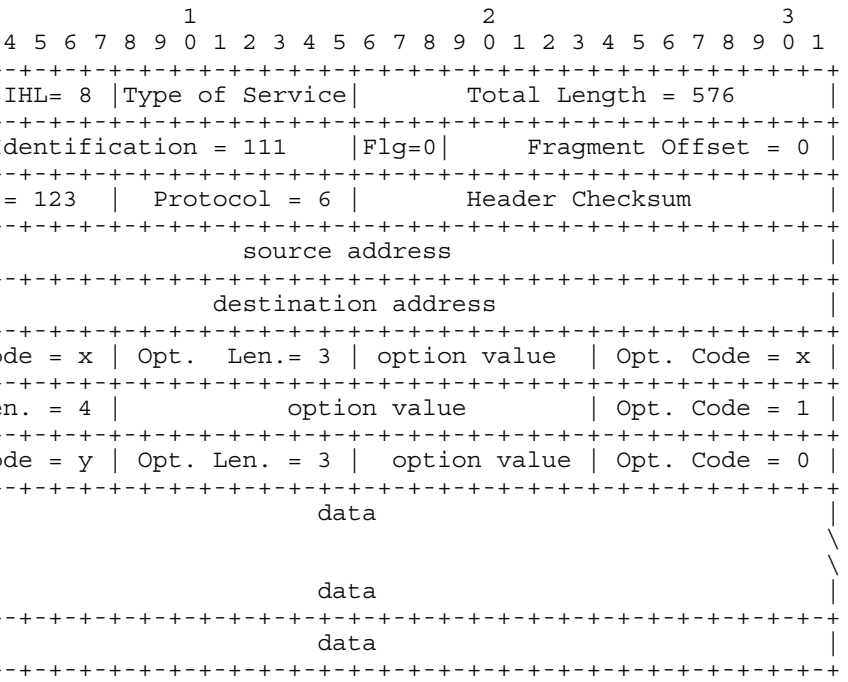


https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 064
Keysight, EX-1045
Page 64

Example 3:

Here, we show an example of a datagram containing options:




Example Internet Datagram

Figure 9.

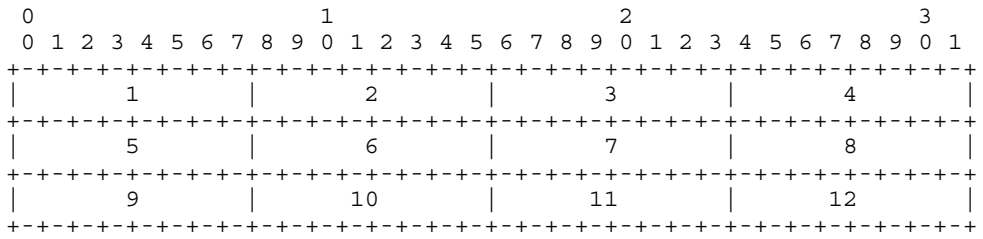


https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 065
Keysight, EX-1045
Page 65

APPENDIX B: Data Transmission Order

The order of transmission of the header and data described in this document is resolved to the octet level. Whenever a diagram shows a group of octets, the order of transmission of those octets is the normal order in which they are read in English. For example, in the following diagram the octets are transmitted in the order they are numbered.




Transmission Order of Bytes

Figure 10.

Whenever an octet represents a numeric quantity the left most bit in the diagram is the high order or most significant bit. That is, the bit labeled 0 is the most significant bit. For example, the following diagram represents the value 170 (decimal).




Significance of Bits

Figure 11.

Similarly, whenever a multi-octet field represents a numeric quantity the left most bit of the whole field is the most significant bit. When a multi-octet quantity is transmitted the most significant octet is transmitted first.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 066
Keysight, EX-1045
Page 66

https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 067
Keysight, EX-1045
Page 67

## GLOSSARY

1822
: BBN Report 1822, "The Specification of the Interconnection of a Host and an IMP". The specification of interface between a host and the ARPANET.

ARPANET leader
: The control information on an ARPANET message at the host-IMP interface.

ARPANET message
: The unit of transmission between a host and an IMP in the ARPANET. The maximum size is about 1012 octets (8096 bits).

ARPANET packet
: A unit of transmission used internally in the ARPANET between IMPs. The maximum size is about 126 octets (1008 bits).

Destination
: The destination address, an internet header field.

DF
: The Don't Fragment bit carried in the flags field.

Flags
: An internet header field carrying various control flags.

Fragment Offset
: This internet header field indicates where in the internet datagram a fragment belongs.

GGP
: Gateway to Gateway Protocol, the protocol used primarily between gateways to control routing and other gateway functions.

header
: Control information at the beginning of a message, segment, datagram, packet or block of data.

ICMP
: Internet Control Message Protocol, implemented in the internet module, the ICMP is used from gateways to hosts and between hosts to report errors and make routing suggestions.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 068
Keysight, EX-1045
Page 68

Identification
An internet header field carrying the identifying value assigned by the sender to aid in assembling the fragments of a datagram.

IHL
The internet header field Internet Header Length is the length of the internet header measured in 32 bit words.

IMP
The Interface Message Processor, the packet switch of the ARPANET.

Internet Address
A four octet (32 bit) source or destination address consisting of a Network field and a Local Address field.

internet datagram
The unit of data exchanged between a pair of internet modules (includes the internet header).

internet fragment
A portion of the data of an internet datagram with an internet header.

Local Address
The address of a host within a network. The actual mapping of an internet local address on to the host addresses in a network is quite general, allowing for many to one mappings.

MF
The More-Fragments Flag carried in the internet header flags field.

module
An implementation, usually in software, of a protocol or other procedure.

more-fragments flag
A flag indicating whether or not this internet datagram contains the end of an internet datagram, carried in the internet header Flags field.

NFB
The Number of Fragment Blocks in a the data portion of an internet fragment. That is, the length of a portion of data measured in 8 octet units.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 069
Keysight, EX-1045
Page 69

octet
An eight bit byte.

Options
The internet header Options field may contain several options, and each option may be several octets in length.

Padding
The internet header Padding field is used to ensure that the data begins on 32 bit word boundary. The padding is zero.

Protocol
In this document, the next higher level protocol identifier, an internet header field.

Rest
The local address portion of an Internet Address.

Source
The source address, an internet header field.

TCP
Transmission Control Protocol: A host-to-host protocol for reliable communication in internet environments.

TCP Segment
The unit of data exchanged between TCP modules (including the TCP header).

TFTP
Trivial File Transfer Protocol: A simple file transfer protocol built on UDP.

Time to Live
An internet header field which indicates the upper bound on how long this internet datagram may exist.

TOS
Type of Service

Total Length
The internet header field Total Length is the length of the datagram in octets including internet header and data.

TTL
Time to Live



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 070
Keysight, EX-1045
Page 70

Type of Service
An internet header field which indicates the type (or quality) of service for this internet datagram.

UDP
User Datagram Protocol: A user level protocol for transaction oriented applications.

User
The user of the internet protocol. This may be a higher level protocol module, an application program, or a gateway program.

Version
The Version field indicates the format of the internet header.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 071
Keysight, EX-1045
Page 71

## REFERENCES

[1] Cerf, V., "The Catenet Model for Internetworking," Information Processing Techniques Office, Defense Advanced Research Projects Agency, IEN 48, July 1978.

[2] Bolt Beranek and Newman, "Specification for the Interconnection of a Host and an IMP," BBN Technical Report 1822, Revised May 1978.

[3] Postel, J., "Internet Control Message Protocol - DARPA Internet Program Protocol Specification," RFC 792, USC/Information Sciences Institute, September 1981.

[4] Shoch, J., "Inter-Network Naming, Addressing, and Routing," COMPCON, IEEE Computer Society, Fall 1978.

[5] Postel, J., "Address Mappings," RFC 796, USC/Information Sciences Institute, September 1981.

[6] Shoch, J., "Packet Fragmentation in Inter-Network Protocols," Computer Networks, v. 3, n. 1, February 1979.

[7] Strazisar, V., "How to Build a Gateway", IEN 109, Bolt Beranek and Newman, August 1979.

[8] Postel, J., "Service Mappings," RFC 795, USC/Information Sciences Institute, September 1981.

[9] Postel, J., "Assigned Numbers," RFC 790, USC/Information Sciences Institute, September 1981.



https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 072
Keysight, EX-1045
Page 72

https://web.archive.org/web/20070204151303/https://tools.ietf.org/html/rfc791

RFC: 793

# TRANSMISSION CONTROL PROTOCOL

## DARPA INTERNET PROGRAM

## PROTOCOL SPECIFICATION

September 1981

**prepared for**

Defense Advanced Research Projects Agency
Information Processing Techniques Office
1400 Wilson Boulevard
Arlington, Virginia 22209

by

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, California 90291

https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 074
**Keysight, EX-1045**
**Page 74**

https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

TABLE OF CONTENTS

PREFACE ..... iii

1. INTRODUCTION ..... 1

1.1 Motivation ..... 1
1.2 Scope ..... 2
1.3 About This Document ..... 2
1.4 Interfaces ..... 3
1.5 Operation ..... 3

2. PHILOSOPHY ..... 7

2.1 Elements of the Internetwork System ..... 7
2.2 Model of Operation ..... 7
2.3 The Host Environment ..... 8
2.4 Interfaces ..... 9
2.5 Relation to Other Protocols ..... 9
2.6 Reliable Communication ..... 9
2.7 Connection Establishment and Clearing ..... 10
2.8 Data Communication ..... 12
2.9 Precedence and Security ..... 13
2.10 Robustness Principle ..... 13

3. FUNCTIONAL SPECIFICATION ..... 15

3.1 Header Format ..... 15
3.2 Terminology ..... 19
3.3 Sequence Numbers ..... 24
3.4 Establishing a connection ..... 30
3.5 Closing a Connection ..... 37
3.6 Precedence and Security ..... 40
3.7 Data Communication ..... 40
3.8 Interfaces ..... 44
3.9 Event Processing ..... 52

GLOSSARY ..... 79

REFERENCES ..... 85



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 076
Keysight, EX-1045
Page 76

https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 077
Keysight, EX-1045
Page 77

## PREFACE

This document describes the DoD Standard Transmission Control Protocol (TCP). There have been nine earlier editions of the ARPA TCP specification on which this standard is based, and the present text draws heavily from them. There have been many contributors to this work both in terms of concepts and in terms of text. This edition clarifies several details and removes the end-of-letter buffer-size adjustments, and redescribes the letter mechanism as a push function.

Jon Postel

Editor



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 078
Keysight, EX-1045
Page 78

RFC: 793
Replaces: RFC 761
IENs: 129, 124, 112, 81,
55, 44, 40, 27, 21, 5

# TRANSMISSION CONTROL PROTOCOL

## DARPA INTERNET PROGRAM
## PROTOCOL SPECIFICATION

## 1. INTRODUCTION

The Transmission Control Protocol (TCP) is intended for use as a highly reliable host-to-host protocol between hosts in packet-switched computer communication networks, and in interconnected systems of such networks.

This document describes the functions to be performed by the Transmission Control Protocol, the program that implements it, and its interface to programs or users that require its services.

### 1.1. Motivation

Computer communication systems are playing an increasingly important role in military, government, and civilian environments. This document focuses its attention primarily on military computer communication requirements, especially robustness in the presence of communication unreliability and availability in the presence of congestion, but many of these problems are found in the civilian and government sector as well.

As strategic and tactical computer communication networks are developed and deployed, it is essential to provide means of interconnecting them and to provide standard interprocess communication protocols which can support a broad range of applications. In anticipation of the need for such standards, the Deputy Undersecretary of Defense for Research and Engineering has declared the Transmission Control Protocol (TCP) described herein to be a basis for DoD-wide inter-process communication protocol standardization.

TCP is a connection-oriented, end-to-end reliable protocol designed to fit into a layered hierarchy of protocols which support multi-network applications. The TCP provides for reliable inter-process communication between pairs of processes in host computers attached to distinct but interconnected computer communication networks. Very few assumptions are made as to the reliability of the communication protocols below the TCP layer. TCP assumes it can obtain a simple, potentially unreliable datagram service from the lower level protocols. In principle, the TCP should be able to operate above a wide spectrum of communication systems ranging from hard-wired connections to packet-switched or circuit-switched networks.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 079
Keysight, EX-1045
Page 79

TCP is based on concepts first described by Cerf and Kahn in [1]. The TCP fits into a layered protocol architecture just above a basic Internet Protocol [2] which provides a way for the TCP to send and receive variable-length segments of information enclosed in internet datagram "envelopes". The internet datagram provides a means for addressing source and destination TCPs in different networks. The internet protocol also deals with any fragmentation or reassembly of the TCP segments required to achieve transport and delivery through multiple networks and interconnecting gateways. The internet protocol also carries information on the precedence, security classification and compartmentation of the TCP segments, so this information can be communicated end-to-end across multiple networks.

Protocol Layering

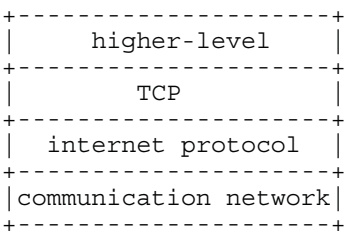



Figure 1

Much of this document is written in the context of TCP implementations which are co-resident with higher level protocols in the host computer. Some computer systems will be connected to networks via front-end computers which house the TCP and internet protocol layers, as well as network specific software. The TCP specification describes an interface to the higher level protocols which appears to be implementable even for the front-end case, as long as a suitable host-to-front end protocol is implemented.

### 1.2. Scope

The TCP is intended to provide a reliable process-to-process communication service in a multinetwork environment. The TCP is intended to be a host-to-host protocol in common use in multiple networks.

### 1.3. About this Document

This document represents a specification of the behavior required of any TCP implementation, both in its interactions with higher level protocols and in its interactions with other TCPs. The rest of this



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 080
Keysight, EX-1045
Page 80

section offers a very brief view of the protocol interfaces and operation. Section 2 summarizes the philosophical basis for the TCP design. Section 3 offers both a detailed description of the actions required of TCP when various events occur (arrival of new segments, user calls, errors, etc.) and the details of the formats of TCP segments.

### 1.4. Interfaces

The TCP interfaces on one side to user or application processes and on the other side to a lower level protocol such as Internet Protocol.

The interface between an application process and the TCP is illustrated in reasonable detail. This interface consists of a set of calls much like the calls an operating system provides to an application process for manipulating files. For example, there are calls to open and close connections and to send and receive data on established connections. It is also expected that the TCP can asynchronously communicate with application programs. Although considerable freedom is permitted to TCP implementors to design interfaces which are appropriate to a particular operating system environment, a minimum functionality is required at the TCP/user interface for any valid implementation.

The interface between TCP and lower level protocol is essentially unspecified except that it is assumed there is a mechanism whereby the two levels can asynchronously pass information to each other. Typically, one expects the lower level protocol to specify this interface. TCP is designed to work in a very general environment of interconnected networks. The lower level protocol which is assumed throughout this document is the Internet Protocol [2].

### 1.5. Operation

As noted above, the primary purpose of the TCP is to provide reliable, securable logical circuit or connection service between pairs of processes. To provide this service on top of a less reliable internet communication system requires facilities in the following areas:

- Basic Data Transfer
- Reliability
- Flow Control
- Multiplexing
- Connections
- Precedence and Security

The basic operation of the TCP in each of these areas is described in the following paragraphs.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 081
Keysight, EX-1045
Page 81

Basic Data Transfer:

The TCP is able to transfer a continuous stream of octets in each direction between its users by packaging some number of octets into segments for transmission through the internet system. In general, the TCPs decide when to block and forward data at their own convenience.

Sometimes users need to be sure that all the data they have submitted to the TCP has been transmitted. For this purpose a push function is defined. To assure that data submitted to a TCP is actually transmitted the sending user indicates that it should be pushed through to the receiving user. A push causes the TCPs to promptly forward and deliver data up to that point to the receiver. The exact push point might not be visible to the receiving user and the push function does not supply a record boundary marker.

Reliability:

The TCP must recover from data that is damaged, lost, duplicated, or delivered out of order by the internet communication system. This is achieved by assigning a sequence number to each octet transmitted, and requiring a positive acknowledgment (ACK) from the receiving TCP. If the ACK is not received within a timeout interval, the data is retransmitted. At the receiver, the sequence numbers are used to correctly order segments that may be received out of order and to eliminate duplicates. Damage is handled by adding a checksum to each segment transmitted, checking it at the receiver, and discarding damaged segments.

As long as the TCPs continue to function properly and the internet system does not become completely partitioned, no transmission errors will affect the correct delivery of data. TCP recovers from internet communication system errors.

Flow Control:

TCP provides a means for the receiver to govern the amount of data sent by the sender. This is achieved by returning a "window" with every ACK indicating a range of acceptable sequence numbers beyond the last segment successfully received. The window indicates an allowed number of octets that the sender may transmit before receiving further permission.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 082
Keysight, EX-1045
Page 82

Multiplexing:

To allow for many processes within a single Host to use TCP communication facilities simultaneously, the TCP provides a set of addresses or ports within each host. Concatenated with the network and host addresses from the internet communication layer, this forms a socket. A pair of sockets uniquely identifies each connection. That is, a socket may be simultaneously used in multiple connections.

The binding of ports to processes is handled independently by each Host. However, it proves useful to attach frequently used processes (e.g., a "logger" or timesharing service) to fixed sockets which are made known to the public. These services can then be accessed through the known addresses. Establishing and learning the port addresses of other processes may involve more dynamic mechanisms.

Connections:

The reliability and flow control mechanisms described above require that TCPs initialize and maintain certain status information for each data stream. The combination of this information, including sockets, sequence numbers, and window sizes, is called a connection. Each connection is uniquely specified by a pair of sockets identifying its two sides.

When two processes wish to communicate, their TCP's must first establish a connection (initialize the status information on each side). When their communication is complete, the connection is terminated or closed to free the resources for other uses.

Since connections must be established between unreliable hosts and over the unreliable internet communication system, a handshake mechanism with clock-based sequence numbers is used to avoid erroneous initialization of connections.

Precedence and Security:

The users of TCP may indicate the security and precedence of their communication. Provision is made for default values to be used when these features are not needed.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 083
Keysight, EX-1045
Page 83

https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 084
Keysight, EX-1045
Page 84

# 2. PHILOSOPHY

## 2.1. Elements of the Internetwork System

The internetwork environment consists of hosts connected to networks which are in turn interconnected via gateways. It is assumed here that the networks may be either local networks (e.g., the ETHERNET) or large networks (e.g., the ARPANET), but in any case are based on packet switching technology. The active agents that produce and consume messages are processes. Various levels of protocols in the networks, the gateways, and the hosts support an interprocess communication system that provides two-way data flow on logical connections between process ports.

The term packet is used generically here to mean the data of one transaction between a host and its network. The format of data blocks exchanged within the a network will generally not be of concern to us.

Hosts are computers attached to a network, and from the communication network's point of view, are the sources and destinations of packets. Processes are viewed as the active elements in host computers (in accordance with the fairly common definition of a process as a program in execution). Even terminals and files or other I/O devices are viewed as communicating with each other through the use of processes. Thus, all communication is viewed as inter-process communication.

Since a process may need to distinguish among several communication streams between itself and another process (or processes), we imagine that each process may have a number of ports through which it communicates with the ports of other processes.

## 2.2. Model of Operation

Processes transmit data by calling on the TCP and passing buffers of data as arguments. The TCP packages the data from these buffers into segments and calls on the internet module to transmit each segment to the destination TCP. The receiving TCP places the data from a segment into the receiving user's buffer and notifies the receiving user. The TCPs include control information in the segments which they use to ensure reliable ordered data transmission.

The model of internet communication is that there is an internet protocol module associated with each TCP which provides an interface to the local network. This internet module packages TCP segments inside internet datagrams and routes these datagrams to a destination internet module or intermediate gateway. To transmit the datagram through the local network, it is embedded in a local network packet.

The packet switches may perform further packaging, fragmentation, or



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 085
Keysight, EX-1045
Page 85

other operations to achieve the delivery of the local packet to the destination internet module.

At a gateway between networks, the internet datagram is "unwrapped" from its local packet and examined to determine through which network the internet datagram should travel next. The internet datagram is then "wrapped" in a local packet suitable to the next network and routed to the next gateway, or to the final destination.

A gateway is permitted to break up an internet datagram into smaller internet datagram fragments if this is necessary for transmission through the next network. To do this, the gateway produces a set of internet datagrams; each carrying a fragment. Fragments may be further broken into smaller fragments at subsequent gateways. The internet datagram fragment format is designed so that the destination internet module can reassemble fragments into internet datagrams.

A destination internet module unwraps the segment from the datagram (after reassembling the datagram, if necessary) and passes it to the destination TCP.

This simple model of the operation glosses over many details. One important feature is the type of service. This provides information to the gateway (or internet module) to guide it in selecting the service parameters to be used in traversing the next network. Included in the type of service information is the precedence of the datagram. Datagrams may also carry security information to permit host and gateways that operate in multilevel secure environments to properly segregate datagrams for security considerations.

### 2.3. The Host Environment

The TCP is assumed to be a module in an operating system. The users access the TCP much like they would access the file system. The TCP may call on other operating system functions, for example, to manage data structures. The actual interface to the network is assumed to be controlled by a device driver module. The TCP does not call on the network device driver directly, but rather calls on the internet datagram protocol module which may in turn call on the device driver.

The mechanisms of TCP do not preclude implementation of the TCP in a front-end processor. However, in such an implementation, a host-to-front-end protocol must provide the functionality to support the type of TCP-user interface described in this document.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 086
Keysight, EX-1045
Page 86

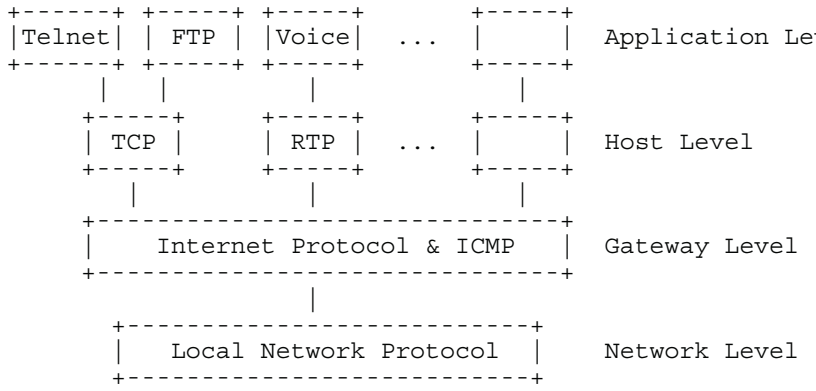
### 2.4. Interfaces

The TCP/user interface provides for calls made by the user on the TCP to OPEN or CLOSE a connection, to SEND or RECEIVE data, or to obtain STATUS about a connection. These calls are like other calls from user programs on the operating system, for example, the calls to open, read from, and close a file.

The TCP/internet interface provides calls to send and receive datagrams addressed to TCP modules in hosts anywhere in the internet system. These calls have parameters for passing the address, type of service, precedence, security, and other control information.

### 2.5. Relation to Other Protocols

The following diagram illustrates the place of the TCP in the protocol hierarchy:




Protocol Relationships

Figure 2.

It is expected that the TCP will be able to support higher level protocols efficiently. It should be easy to interface higher level protocols like the ARPANET Telnet or AUTODIN II THP to the TCP.

### 2.6. Reliable Communication

A stream of data sent on a TCP connection is delivered reliably and in order at the destination.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 087
Keysight, EX-1045
Page 87

Transmission is made reliable via the use of sequence numbers and acknowledgments. Conceptually, each octet of data is assigned a sequence number. The sequence number of the first octet of data in a segment is transmitted with that segment and is called the segment sequence number. Segments also carry an acknowledgment number which is the sequence number of the next expected data octet of transmissions in the reverse direction. When the TCP transmits a segment containing data, it puts a copy on a retransmission queue and starts a timer; when the acknowledgment for that data is received, the segment is deleted from the queue. If the acknowledgment is not received before the timer runs out, the segment is retransmitted.

An acknowledgment by TCP does not guarantee that the data has been delivered to the end user, but only that the receiving TCP has taken the responsibility to do so.

To govern the flow of data between TCPs, a flow control mechanism is employed. The receiving TCP reports a "window" to the sending TCP. This window specifies the number of octets, starting with the acknowledgment number, that the receiving TCP is currently prepared to receive.

### 2.7. Connection Establishment and Clearing

To identify the separate data streams that a TCP may handle, the TCP provides a port identifier. Since port identifiers are selected independently by each TCP they might not be unique. To provide for unique addresses within each TCP, we concatenate an internet address identifying the TCP with a port identifier to create a socket which will be unique throughout all networks connected together.

A connection is fully specified by the pair of sockets at the ends. A local socket may participate in many connections to different foreign sockets. A connection can be used to carry data in both directions, that is, it is "full duplex".

TCPs are free to associate ports with processes however they choose. However, several basic concepts are necessary in any implementation. There must be well-known sockets which the TCP associates only with the "appropriate" processes by some means. We envision that processes may "own" ports, and that processes can initiate connections only on the ports they own. (Means for implementing ownership is a local issue, but we envision a Request Port user command, or a method of uniquely allocating a group of ports to a given process, e.g., by associating the high order bits of a port name with a given process.)

A connection is specified in the OPEN call by the local port and foreign socket arguments. In return, the TCP supplies a (short) local



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 088
Keysight, EX-1045
Page 88

connection name by which the user refers to the connection in subsequent calls. There are several things that must be remembered about a connection. To store this information we imagine that there is a data structure called a Transmission Control Block (TCB). One implementation strategy would have the local connection name be a pointer to the TCB for this connection. The OPEN call also specifies whether the connection establishment is to be actively pursued, or to be passively waited for.

A passive OPEN request means that the process wants to accept incoming connection requests rather than attempting to initiate a connection. Often the process requesting a passive OPEN will accept a connection request from any caller. In this case a foreign socket of all zeros is used to denote an unspecified socket. Unspecified foreign sockets are allowed only on passive OPENs.

A service process that wished to provide services for unknown other processes would issue a passive OPEN request with an unspecified foreign socket. Then a connection could be made with any process that requested a connection to this local socket. It would help if this local socket were known to be associated with this service.

Well-known sockets are a convenient mechanism for a priori associating a socket address with a standard service. For instance, the "Telnet-Server" process is permanently assigned to a particular socket, and other sockets are reserved for File Transfer, Remote Job Entry, Text Generator, Echoer, and Sink processes (the last three being for test purposes). A socket address might be reserved for access to a "Look-Up" service which would return the specific socket at which a newly created service would be provided. The concept of a well-known socket is part of the TCP specification, but the assignment of sockets to services is outside this specification. (See [4].)

Processes can issue passive OPENs and wait for matching active OPENs from other processes and be informed by the TCP when connections have been established. Two processes which issue active OPENs to each other at the same time will be correctly connected. This flexibility is critical for the support of distributed computing in which components act asynchronously with respect to each other.

There are two principal cases for matching the sockets in the local passive OPENs and an foreign active OPENs. In the first case, the local passive OPENs has fully specified the foreign socket. In this case, the match must be exact. In the second case, the local passive OPENs has left the foreign socket unspecified. In this case, any foreign socket is acceptable as long as the local sockets match. Other possibilities include partially restricted matches.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 089
Keysight, EX-1045
Page 89

If there are several pending passive OPENs (recorded in TCBs) with the same local socket, an foreign active OPEN will be matched to a TCB with the specific foreign socket in the foreign active OPEN, if such a TCB exists, before selecting a TCB with an unspecified foreign socket.

The procedures to establish connections utilize the synchronize (SYN) control flag and involves an exchange of three messages. This exchange has been termed a three-way hand shake [3].

A connection is initiated by the rendezvous of an arriving segment containing a SYN and a waiting TCB entry each created by a user OPEN command. The matching of local and foreign sockets determines when a connection has been initiated. The connection becomes "established" when sequence numbers have been synchronized in both directions.

The clearing of a connection also involves the exchange of segments, in this case carrying the FIN control flag.

### 2.8. Data Communication

The data that flows on a connection may be thought of as a stream of octets. The sending user indicates in each SEND call whether the data in that call (and any preceeding calls) should be immediately pushed through to the receiving user by the setting of the PUSH flag.

A sending TCP is allowed to collect data from the sending user and to send that data in segments at its own convenience, until the push function is signaled, then it must send all unsent data. When a receiving TCP sees the PUSH flag, it must not wait for more data from the sending TCP before passing the data to the receiving process.

There is no necessary relationship between push functions and segment boundaries. The data in any particular segment may be the result of a single SEND call, in whole or part, or of multiple SEND calls.

The purpose of push function and the PUSH flag is to push data through from the sending user to the receiving user. It does not provide a record service.

There is a coupling between the push function and the use of buffers of data that cross the TCP/user interface. Each time a PUSH flag is associated with data placed into the receiving user's buffer, the buffer is returned to the user for processing even if the buffer is not filled. If data arrives that fills the user's buffer before a PUSH is seen, the data is passed to the user in buffer size units.

TCP also provides a means to communicate to the receiver of data that at some point further along in the data stream than the receiver is



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 090
Keysight, EX-1045
Page 90

currently reading there is urgent data. TCP does not attempt to define what the user specifically does upon being notified of pending urgent data, but the general notion is that the receiving process will take action to process the urgent data quickly.

**2.9. Precedence and Security**

The TCP makes use of the internet protocol type of service field and security option to provide precedence and security on a per connection basis to TCP users. Not all TCP modules will necessarily function in a multilevel secure environment; some may be limited to unclassified use only, and others may operate at only one security level and compartment. Consequently, some TCP implementations and services to users may be limited to a subset of the multilevel secure case.

TCP modules which operate in a multilevel secure environment must properly mark outgoing segments with the security, compartment, and precedence. Such TCP modules must also provide to their users or higher level protocols such as Telnet or THP an interface to allow them to specify the desired security level, compartment, and precedence of connections.

**2.10. Robustness Principle**

TCP implementations will follow a general principle of robustness: be conservative in what you do, be liberal in what you accept from others.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 091
Keysight, EX-1045
Page 91

https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

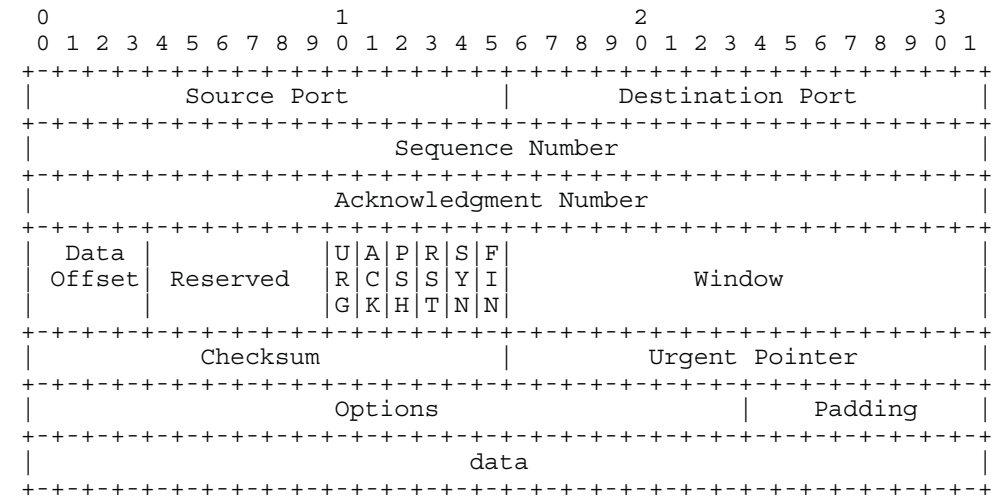
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 092
Keysight, EX-1045
Page 92

# 3. FUNCTIONAL SPECIFICATION

## 3.1. Header Format

TCP segments are sent as internet datagrams. The Internet Protocol header carries several information fields, including the source and destination host addresses [2]. A TCP header follows the internet header, supplying information specific to the TCP protocol. This division allows for the existence of host level protocols other than TCP.

TCP Header Format




TCP Header Format

Note that one tick mark represents one bit position.

Figure 3.

Source Port: 16 bits

The source port number.

Destination Port: 16 bits

The destination port number.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 093
Keysight, EX-1045
Page 93

Sequence Number: 32 bits

The sequence number of the first data octet in this segment (except when SYN is present). If SYN is present the sequence number is the initial sequence number (ISN) and the first data octet is ISN+1.

Acknowledgment Number: 32 bits

If the ACK control bit is set this field contains the value of the next sequence number the sender of the segment is expecting to receive. Once a connection is established this is always sent.

Data Offset: 4 bits

The number of 32 bit words in the TCP Header. This indicates where the data begins. The TCP header (even one including options) is an integral number of 32 bits long.

Reserved: 6 bits

Reserved for future use. Must be zero.

Control Bits: 6 bits (from left to right):

URG: Urgent Pointer field significant
ACK: Acknowledgment field significant
PSH: Push Function
RST: Reset the connection
SYN: Synchronize sequence numbers
FIN: No more data from sender

Window: 16 bits

The number of data octets beginning with the one indicated in the acknowledgment field which the sender of this segment is willing to accept.

Checksum: 16 bits

The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header and text. If a segment contains an odd number of header and text octets to be checksummed, the last octet is padded on the right with zeros to form a 16 bit word for checksum purposes. The pad is not transmitted as part of the segment. While computing the checksum, the checksum field itself is replaced with zeros.

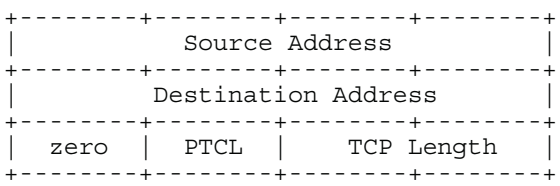
The checksum also covers a 96 bit pseudo header conceptually



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 094
Keysight, EX-1045
Page 94

prefixed to the TCP header. This pseudo header contains the Source Address, the Destination Address, the Protocol, and TCP length. This gives the TCP protection against misrouted segments. This information is carried in the Internet Protocol and is transferred across the TCP/Network interface in the arguments or results of calls by the TCP on the IP.




The TCP Length is the TCP header length plus the data length in octets (this is not an explicitly transmitted quantity, but is computed), and it does not count the 12 octets of the pseudo header.

Urgent Pointer: 16 bits

This field communicates the current value of the urgent pointer as a positive offset from the sequence number in this segment. The urgent pointer points to the sequence number of the octet following the urgent data. This field is only be interpreted in segments with the URG control bit set.

Options: variable

Options may occupy space at the end of the TCP header and are a multiple of 8 bits in length. All options are included in the checksum. An option may begin on any octet boundary. There are two cases for the format of an option:

Case 1: A single octet of option-kind.

Case 2: An octet of option-kind, an octet of option-length, and the actual option-data octets.

The option-length counts the two octets of option-kind and option-length as well as the option-data octets.

Note that the list of options may be shorter than the data offset field might imply. The content of the header beyond the End-of-Option option must be header padding (i.e., zero).

A TCP must implement all options.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 095
Keysight, EX-1045
Page 95

Currently defined options include (kind indicated in octal):

| Kind | Length | Meaning |
|---|---|---|
| 0 | - | End of option list. |
| 1 | - | No-Operation. |
| 2 | 4 | Maximum Segment Size. |

Specific Option Definitions

End of Option List



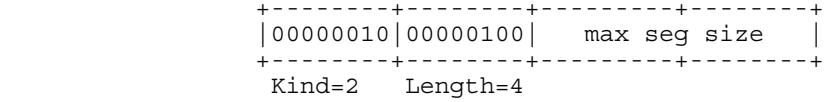

This option code indicates the end of the option list. This might not coincide with the end of the TCP header according to the Data Offset field. This is used at the end of all options, not the end of each option, and need only be used if the end of the options would not otherwise coincide with the end of the TCP header.

No-Operation




This option code may be used between options, for example, to align the beginning of a subsequent option on a word boundary. There is no guarantee that senders will use this option, so receivers must be prepared to process options even if they do not begin on a word boundary.

Maximum Segment Size






https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 096
Keysight, EX-1045
Page 96

Maximum Segment Size Option Data: 16 bits

If this option is present, then it communicates the maximum receive segment size at the TCP which sends this segment. This field must only be sent in the initial connection request (i.e., in segments with the SYN control bit set). If this option is not used, any segment size is allowed.

Padding: variable

The TCP header padding is used to ensure that the TCP header ends and data begins on a 32 bit boundary. The padding is composed of zeros.

### 3.2. Terminology

Before we can discuss very much about the operation of the TCP we need to introduce some detailed terminology. The maintenance of a TCP connection requires the remembering of several variables. We conceive of these variables being stored in a connection record called a Transmission Control Block or TCB. Among the variables stored in the TCB are the local and remote socket numbers, the security and precedence of the connection, pointers to the user's send and receive buffers, pointers to the retransmit queue and to the current segment. In addition several variables relating to the send and receive sequence numbers are stored in the TCB.

Send Sequence Variables

```
SND.UNA - send unacknowledged
SND.NXT - send next
SND.WND - send window
SND.UP  - send urgent pointer
SND.WL1 - segment sequence number used for last window update
SND.WL2 - segment acknowledgment number used for last window
          update
ISS     - initial send sequence number
```

Receive Sequence Variables

```
RCV.NXT - receive next
RCV.WND - receive window
RCV.UP  - receive urgent pointer
IRS     - initial receive sequence number
```

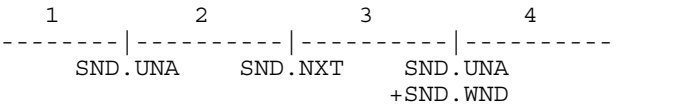


https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 097
Keysight, EX-1045
Page 97

The following diagrams may help to relate some of these variables to the sequence space.

Send Sequence Space



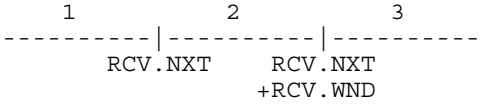

1 - old sequence numbers which have been acknowledged
2 - sequence numbers of unacknowledged data
3 - sequence numbers allowed for new data transmission
4 - future sequence numbers which are not yet allowed

Send Sequence Space

Figure 4.

The send window is the portion of the sequence space labeled 3 in figure 4.

Receive Sequence Space




1 - old sequence numbers which have been acknowledged
2 - sequence numbers allowed for new reception
3 - future sequence numbers which are not yet allowed

Receive Sequence Space

Figure 5.

The receive window is the portion of the sequence space labeled 2 in figure 5.

There are also some variables used frequently in the discussion that take their values from the fields of the current segment.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 098
Keysight, EX-1045
Page 98

Current Segment Variables

SEG.SEQ - segment sequence number
SEG.ACK - segment acknowledgment number
SEG.LEN - segment length
SEG.WND - segment window
SEG.UP - segment urgent pointer
SEG.PRC - segment precedence value

A connection progresses through a series of states during its lifetime. The states are: LISTEN, SYN-SENT, SYN-RECEIVED, ESTABLISHED, FIN-WAIT-1, FIN-WAIT-2, CLOSE-WAIT, CLOSING, LAST-ACK, TIME-WAIT, and the fictional state CLOSED. CLOSED is fictional because it represents the state when there is no TCB, and therefore, no connection. Briefly the meanings of the states are:

LISTEN - represents waiting for a connection request from any remote TCP and port.

SYN-SENT - represents waiting for a matching connection request after having sent a connection request.

SYN-RECEIVED - represents waiting for a confirming connection request acknowledgment after having both received and sent a connection request.

ESTABLISHED - represents an open connection, data received can be delivered to the user. The normal state for the data transfer phase of the connection.

FIN-WAIT-1 - represents waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent.

FIN-WAIT-2 - represents waiting for a connection termination request from the remote TCP.

CLOSE-WAIT - represents waiting for a connection termination request from the local user.

CLOSING - represents waiting for a connection termination request acknowledgment from the remote TCP.

LAST-ACK - represents waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request).



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 099
Keysight, EX-1045
Page 99

TIME-WAIT - represents waiting for enough time to pass to be sure the remote TCP received the acknowledgment of its connection termination request.

CLOSED - represents no connection state at all.

A TCP connection progresses from one state to another in response to events. The events are the user calls, OPEN, SEND, RECEIVE, CLOSE, ABORT, and STATUS; the incoming segments, particularly those containing the SYN, ACK, RST and FIN flags; and timeouts.

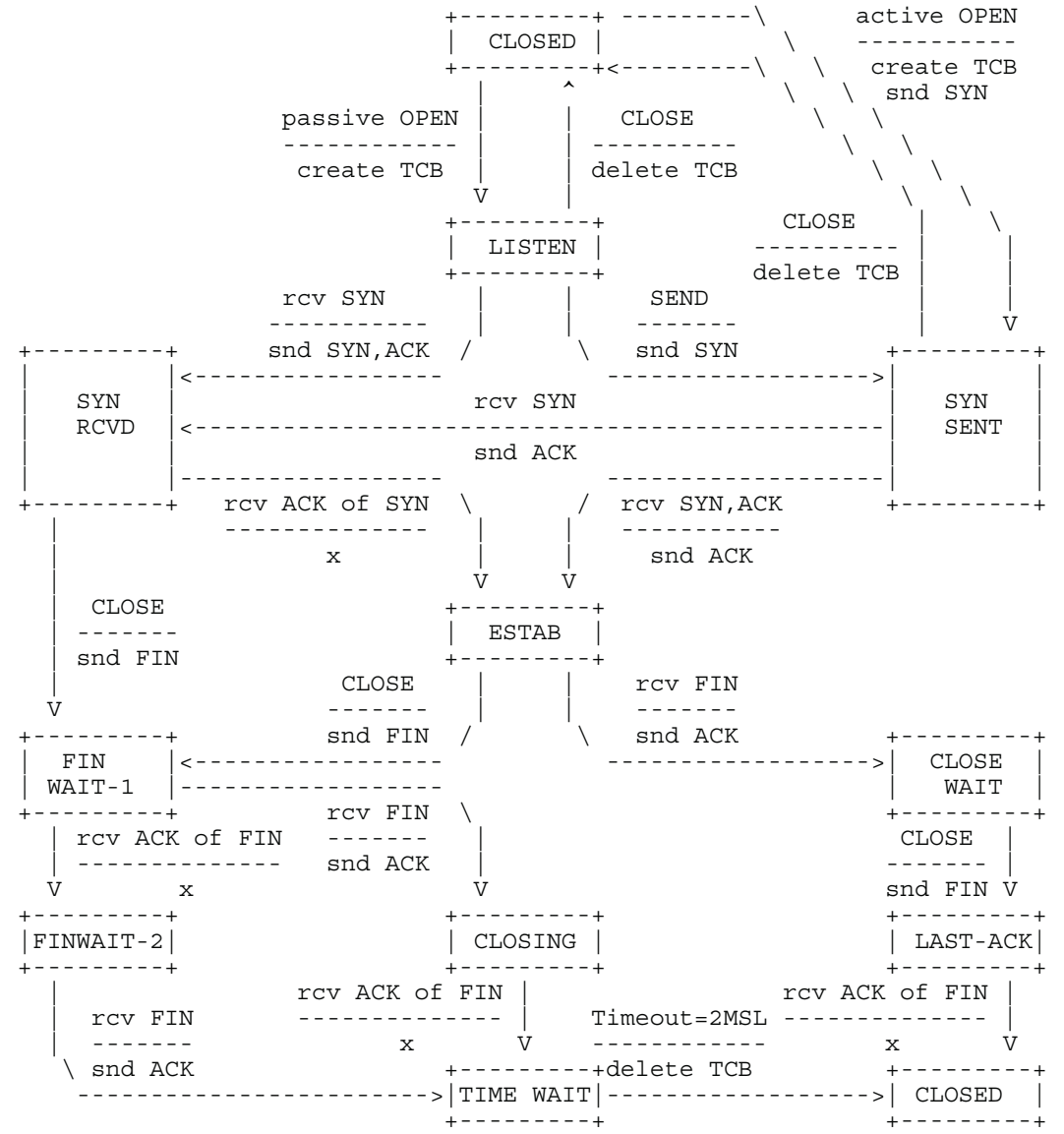
The state diagram in figure 6 illustrates only state changes, together with the causing events and resulting actions, but addresses neither error conditions nor actions which are not connected with state changes. In a later section, more detail is offered with respect to the reaction of the TCP to events.

NOTE BENE: this diagram is only a summary and must not be taken as the total specification.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 100
Keysight, EX-1045
Page 100




TCP Connection State Diagram
Figure 6.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 101
Keysight, EX-1045
Page 101

### 3.3. Sequence Numbers

A fundamental notion in the design is that every octet of data sent over a TCP connection has a sequence number. Since every octet is sequenced, each of them can be acknowledged. The acknowledgment mechanism employed is cumulative so that an acknowledgment of sequence number X indicates that all octets up to but not including X have been received. This mechanism allows for straight-forward duplicate detection in the presence of retransmission. Numbering of octets within a segment is that the first data octet immediately following the header is the lowest numbered, and the following octets are numbered consecutively.

It is essential to remember that the actual sequence number space is finite, though very large. This space ranges from 0 to 2**32 - 1. Since the space is finite, all arithmetic dealing with sequence numbers must be performed modulo 2**32. This unsigned arithmetic preserves the relationship of sequence numbers as they cycle from 2**32 - 1 to 0 again. There are some subtleties to computer modulo arithmetic, so great care should be taken in programming the comparison of such values. The symbol "=<" means "less than or equal" (modulo 2**32).

The typical kinds of sequence number comparisons which the TCP must perform include:

(a) Determining that an acknowledgment refers to some sequence number sent but not yet acknowledged.

(b) Determining that all sequence numbers occupied by a segment have been acknowledged (e.g., to remove the segment from a retransmission queue).

(c) Determining that an incoming segment contains sequence numbers which are expected (i.e., that the segment "overlaps" the receive window).



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 102
Keysight, EX-1045
Page 102

In response to sending data the TCP will receive acknowledgments. The following comparisons are needed to process the acknowledgments.

SND.UNA = oldest unacknowledged sequence number

SND.NXT = next sequence number to be sent

SEG.ACK = acknowledgment from the receiving TCP (next sequence number expected by the receiving TCP)

SEG.SEQ = first sequence number of a segment

SEG.LEN = the number of octets occupied by the data in the segment (counting SYN and FIN)

SEG.SEQ+SEG.LEN-1 = last sequence number of a segment

A new acknowledgment (called an "acceptable ack"), is one for which the inequality below holds:

SND.UNA < SEG.ACK =< SND.NXT

A segment on the retransmission queue is fully acknowledged if the sum of its sequence number and length is less or equal than the acknowledgment value in the incoming segment.

When data is received the following comparisons are needed:

RCV.NXT = next sequence number expected on an incoming segments, and is the left or lower edge of the receive window

RCV.NXT+RCV.WND-1 = last sequence number expected on an incoming segment, and is the right or upper edge of the receive window

SEG.SEQ = first sequence number occupied by the incoming segment

SEG.SEQ+SEG.LEN-1 = last sequence number occupied by the incoming segment

A segment is judged to occupy a portion of valid receive sequence space if

RCV.NXT =< SEG.SEQ < RCV.NXT+RCV.WND

or

RCV.NXT =< SEG.SEQ+SEG.LEN-1 < RCV.NXT+RCV.WND



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 103
Keysight, EX-1045
Page 103

The first part of this test checks to see if the beginning of the segment falls in the window, the second part of the test checks to see if the end of the segment falls in the window; if the segment passes either part of the test it contains data in the window.

Actually, it is a little more complicated than this. Due to zero windows and zero length segments, we have four cases for the acceptability of an incoming segment:

| Segment Length | Receive Window | Test |
|---|---|---|
| 0 | 0 | SEG.SEQ = RCV.NXT |
| 0 | >0 | RCV.NXT =< SEG.SEQ < RCV.NXT+RCV.WND |
| >0 | 0 | not acceptable |
| >0 | >0 | RCV.NXT =< SEG.SEQ < RCV.NXT+RCV.WND<br>or RCV.NXT =< SEG.SEQ+SEG.LEN-1 < RCV.NXT+RCV.WND |

Note that when the receive window is zero no segments should be acceptable except ACK segments. Thus, it is be possible for a TCP to maintain a zero receive window while transmitting data and receiving ACKs. However, even when the receive window is zero, a TCP must process the RST and URG fields of all incoming segments.

We have taken advantage of the numbering scheme to protect certain control information as well. This is achieved by implicitly including some control flags in the sequence space so they can be retransmitted and acknowledged without confusion (i.e., one and only one copy of the control will be acted upon). Control information is not physically carried in the segment data space. Consequently, we must adopt rules for implicitly assigning sequence numbers to control. The SYN and FIN are the only controls requiring this protection, and these controls are used only at connection opening and closing. For sequence number purposes, the SYN is considered to occur before the first actual data octet of the segment in which it occurs, while the FIN is considered to occur after the last actual data octet in a segment in which it occurs. The segment length (SEG.LEN) includes both data and sequence space occupying controls. When a SYN is present then SEG.SEQ is the sequence number of the SYN.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 104
Keysight, EX-1045
Page 104

Initial Sequence Number Selection

The protocol places no restriction on a particular connection being used over and over again. A connection is defined by a pair of sockets. New instances of a connection will be referred to as incarnations of the connection. The problem that arises from this is -- "how does the TCP identify duplicate segments from previous incarnations of the connection?" This problem becomes apparent if the connection is being opened and closed in quick succession, or if the connection breaks with loss of memory and is then reestablished.

To avoid confusion we must prevent segments from one incarnation of a connection from being used while the same sequence numbers may still be present in the network from an earlier incarnation. We want to assure this, even if a TCP crashes and loses all knowledge of the sequence numbers it has been using. When new connections are created, an initial sequence number (ISN) generator is employed which selects a new 32 bit ISN. The generator is bound to a (possibly fictitious) 32 bit clock whose low order bit is incremented roughly every 4 microseconds. Thus, the ISN cycles approximately every 4.55 hours. Since we assume that segments will stay in the network no more than the Maximum Segment Lifetime (MSL) and that the MSL is less than 4.55 hours we can reasonably assume that ISN's will be unique.

For each connection there is a send sequence number and a receive sequence number. The initial send sequence number (ISS) is chosen by the data sending TCP, and the initial receive sequence number (IRS) is learned during the connection establishing procedure.

For a connection to be established or initialized, the two TCPs must synchronize on each other's initial sequence numbers. This is done in an exchange of connection establishing segments carrying a control bit called "SYN" (for synchronize) and the initial sequence numbers. As a shorthand, segments carrying the SYN bit are also called "SYNs". Hence, the solution requires a suitable mechanism for picking an initial sequence number and a slightly involved handshake to exchange the ISN's.

The synchronization requires each side to send it's own initial sequence number and to receive a confirmation of it in acknowledgment from the other side. Each side must also receive the other side's initial sequence number and send a confirming acknowledgment.

1) A --> B SYN my sequence number is X
2) A <-- B ACK your sequence number is X
3) A <-- B SYN my sequence number is Y
4) A --> B ACK your sequence number is Y



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 105
Keysight, EX-1045
Page 105

Because steps 2 and 3 can be combined in a single message this is called the three way (or three message) handshake.

A three way handshake is necessary because sequence numbers are not tied to a global clock in the network, and TCPs may have different mechanisms for picking the ISN's. The receiver of the first SYN has no way of knowing whether the segment was an old delayed one or not, unless it remembers the last sequence number used on the connection (which is not always possible), and so it must ask the sender to verify this SYN. The three way handshake and the advantages of a clock-driven scheme are discussed in [3].

Knowing When to Keep Quiet

To be sure that a TCP does not create a segment that carries a sequence number which may be duplicated by an old segment remaining in the network, the TCP must keep quiet for a maximum segment lifetime (MSL) before assigning any sequence numbers upon starting up or recovering from a crash in which memory of sequence numbers in use was lost. For this specification the MSL is taken to be 2 minutes. This is an engineering choice, and may be changed if experience indicates it is desirable to do so. Note that if a TCP is reinitialized in some sense, yet retains its memory of sequence numbers in use, then it need not wait at all; it must only be sure to use sequence numbers larger than those recently used.

The TCP Quiet Time Concept

This specification provides that hosts which "crash" without retaining any knowledge of the last sequence numbers transmitted on each active (i.e., not closed) connection shall delay emitting any TCP segments for at least the agreed Maximum Segment Lifetime (MSL) in the internet system of which the host is a part. In the paragraphs below, an explanation for this specification is given. TCP implementors may violate the "quiet time" restriction, but only at the risk of causing some old data to be accepted as new or new data rejected as old duplicated by some receivers in the internet system.

TCPs consume sequence number space each time a segment is formed and entered into the network output queue at a source host. The duplicate detection and sequencing algorithm in the TCP protocol relies on the unique binding of segment data to sequence space to the extent that sequence numbers will not cycle through all 2**32 values before the segment data bound to those sequence numbers has been delivered and acknowledged by the receiver and all duplicate copies of the segments have "drained" from the internet. Without such an assumption, two distinct TCP segments could conceivably be



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 106
Keysight, EX-1045
Page 106

assigned the same or overlapping sequence numbers, causing confusion at the receiver as to which data is new and which is old. Remember that each segment is bound to as many consecutive sequence numbers as there are octets of data in the segment.

Under normal conditions, TCPs keep track of the next sequence number to emit and the oldest awaiting acknowledgment so as to avoid mistakenly using a sequence number over before its first use has been acknowledged. This alone does not guarantee that old duplicate data is drained from the net, so the sequence space has been made very large to reduce the probability that a wandering duplicate will cause trouble upon arrival. At 2 megabits/sec. it takes 4.5 hours to use up 2**32 octets of sequence space. Since the maximum segment lifetime in the net is not likely to exceed a few tens of seconds, this is deemed ample protection for foreseeable nets, even if data rates escalate to l0's of megabits/sec. At 100 megabits/sec, the cycle time is 5.4 minutes which may be a little short, but still within reason.

The basic duplicate detection and sequencing algorithm in TCP can be defeated, however, if a source TCP does not have any memory of the sequence numbers it last used on a given connection. For example, if the TCP were to start all connections with sequence number 0, then upon crashing and restarting, a TCP might re-form an earlier connection (possibly after half-open connection resolution) and emit packets with sequence numbers identical to or overlapping with packets still in the network which were emitted on an earlier incarnation of the same connection. In the absence of knowledge about the sequence numbers used on a particular connection, the TCP specification recommends that the source delay for MSL seconds before emitting segments on the connection, to allow time for segments from the earlier connection incarnation to drain from the system.

Even hosts which can remember the time of day and used it to select initial sequence number values are not immune from this problem (i.e., even if time of day is used to select an initial sequence number for each new connection incarnation).

Suppose, for example, that a connection is opened starting with sequence number S. Suppose that this connection is not used much and that eventually the initial sequence number function (ISN(t)) takes on a value equal to the sequence number, say S1, of the last segment sent by this TCP on a particular connection. Now suppose, at this instant, the host crashes, recovers, and establishes a new incarnation of the connection. The initial sequence number chosen is S1 = ISN(t) -- last used sequence number on old incarnation of connection! If the recovery occurs quickly enough, any old



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 107
Keysight, EX-1045
Page 107

duplicates in the net bearing sequence numbers in the neighborhood of S1 may arrive and be treated as new packets by the receiver of the new incarnation of the connection.

The problem is that the recovering host may not know for how long it crashed nor does it know whether there are still old duplicates in the system from earlier connection incarnations.

One way to deal with this problem is to deliberately delay emitting segments for one MSL after recovery from a crash- this is the "quite time" specification. Hosts which prefer to avoid waiting are willing to risk possible confusion of old and new packets at a given destination may choose not to wait for the "quite time". Implementors may provide TCP users with the ability to select on a connection by connection basis whether to wait after a crash, or may informally implement the "quite time" for all connections. Obviously, even where a user selects to "wait," this is not necessary after the host has been "up" for at least MSL seconds.

To summarize: every segment emitted occupies one or more sequence numbers in the sequence space, the numbers occupied by a segment are "busy" or "in use" until MSL seconds have passed, upon crashing a block of space-time is occupied by the octets of the last emitted segment, if a new connection is started too soon and uses any of the sequence numbers in the space-time footprint of the last segment of the previous connection incarnation, there is a potential sequence number overlap area which could cause confusion at the receiver.

### 3.4. Establishing a connection

The "three-way handshake" is the procedure used to establish a connection. This procedure normally is initiated by one TCP and responded to by another TCP. The procedure also works if two TCP simultaneously initiate the procedure. When simultaneous attempt occurs, each TCP receives a "SYN" segment which carries no acknowledgment after it has sent a "SYN". Of course, the arrival of an old duplicate "SYN" segment can potentially make it appear, to the recipient, that a simultaneous connection initiation is in progress. Proper use of "reset" segments can disambiguate these cases.

Several examples of connection initiation follow. Although these examples do not show connection synchronization using data-carrying segments, this is perfectly legitimate, so long as the receiving TCP doesn't deliver the data to the user until it is clear the data is valid (i.e., the data must be buffered at the receiver until the connection reaches the ESTABLISHED state). The three-way handshake reduces the possibility of false connections. It is the



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 108
Keysight, EX-1045
Page 108

implementation of a trade-off between memory and messages to provide information for this checking.

The simplest three-way handshake is shown in figure 7 below. The figures should be interpreted in the following way. Each line is numbered for reference purposes. Right arrows (-->) indicate departure of a TCP segment from TCP A to TCP B, or arrival of a segment at B from A. Left arrows (<--), indicate the reverse. Ellipsis (...) indicates a segment which is still in the network (delayed). An "XXX" indicates a segment which is lost or rejected. Comments appear in parentheses. TCP states represent the state AFTER the departure or arrival of the segment (whose contents are shown in the center of each line). Segment contents are shown in abbreviated form, with sequence number, control flags, and ACK field. Other fields such as window, addresses, lengths, and text have been left out in the interest of clarity.

```
    TCP A                                                TCP B

1.  CLOSED                                               LISTEN

2.  SYN-SENT    --> <SEQ=100><CTL=SYN>               --> SYN-RECEIVED

3.  ESTABLISHED <-- <SEQ=300><ACK=101><CTL=SYN,ACK>  <-- SYN-RECEIVED

4.  ESTABLISHED --> <SEQ=101><ACK=301><CTL=ACK>       --> ESTABLISHED

5.  ESTABLISHED --> <SEQ=101><ACK=301><CTL=ACK><DATA> --> ESTABLISHED
```

Basic 3-Way Handshake for Connection Synchronization

Figure 7.

In line 2 of figure 7, TCP A begins by sending a SYN segment indicating that it will use sequence numbers starting with sequence number 100. In line 3, TCP B sends a SYN and acknowledges the SYN it received from TCP A. Note that the acknowledgment field indicates TCP B is now expecting to hear sequence 101, acknowledging the SYN which occupied sequence 100.

At line 4, TCP A responds with an empty segment containing an ACK for TCP B's SYN; and in line 5, TCP A sends some data. Note that the sequence number of the segment in line 5 is the same as in line 4 because the ACK does not occupy sequence number space (if it did, we would wind up ACKing ACK's!).



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 109
Keysight, EX-1045
Page 109

Simultaneous initiation is only slightly more complex, as is shown in figure 8. Each TCP cycles from CLOSED to SYN-SENT to SYN-RECEIVED to ESTABLISHED.

```
    TCP A                                                TCP B

1.  CLOSED                                               CLOSED

2.  SYN-SENT     --> <SEQ=100><CTL=SYN>              ...

3.  SYN-RECEIVED <-- <SEQ=300><CTL=SYN>              <-- SYN-SENT

4.               ... <SEQ=100><CTL=SYN>              --> SYN-RECEIVED

5.  SYN-RECEIVED --> <SEQ=100><ACK=301><CTL=SYN,ACK> ...

6.  ESTABLISHED  <-- <SEQ=300><ACK=101><CTL=SYN,ACK> <-- SYN-RECEIVED

7.               ... <SEQ=101><ACK=301><CTL=ACK>     --> ESTABLISHED
```

Simultaneous Connection Synchronization

Figure 8.

The principle reason for the three-way handshake is to prevent old duplicate connection initiations from causing confusion. To deal with this, a special control message, reset, has been devised. If the receiving TCP is in a non-synchronized state (i.e., SYN-SENT, SYN-RECEIVED), it returns to LISTEN on receiving an acceptable reset. If the TCP is in one of the synchronized states (ESTABLISHED, FIN-WAIT-1, FIN-WAIT-2, CLOSE-WAIT, CLOSING, LAST-ACK, TIME-WAIT), it aborts the connection and informs its user. We discuss this latter case under "half-open" connections below.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 110
Keysight, EX-1045
Page 110

```
    TCP A                                                TCP B

1.  CLOSED                                               LISTEN

2.  SYN-SENT    --> <SEQ=100><CTL=SYN>               ...

3.  (duplicate) ... <SEQ=90><CTL=SYN>               --> SYN-RECEIVED

4.  SYN-SENT    <-- <SEQ=300><ACK=91><CTL=SYN,ACK>  <-- SYN-RECEIVED

5.  SYN-SENT    --> <SEQ=91><CTL=RST>               --> LISTEN

6.              ... <SEQ=100><CTL=SYN>               --> SYN-RECEIVED

7.  SYN-SENT    <-- <SEQ=400><ACK=101><CTL=SYN,ACK>  <-- SYN-RECEIVED

8.  ESTABLISHED --> <SEQ=101><ACK=401><CTL=ACK>      --> ESTABLISHED
```

Recovery from Old Duplicate SYN

Figure 9.

As a simple example of recovery from old duplicates, consider figure 9. At line 3, an old duplicate SYN arrives at TCP B. TCP B cannot tell that this is an old duplicate, so it responds normally (line 4). TCP A detects that the ACK field is incorrect and returns a RST (reset) with its SEQ field selected to make the segment believable. TCP B, on receiving the RST, returns to the LISTEN state. When the original SYN (pun intended) finally arrives at line 6, the synchronization proceeds normally. If the SYN at line 6 had arrived before the RST, a more complex exchange might have occurred with RST's sent in both directions.

Half-Open Connections and Other Anomalies

An established connection is said to be "half-open" if one of the TCPs has closed or aborted the connection at its end without the knowledge of the other, or if the two ends of the connection have become desynchronized owing to a crash that resulted in loss of memory. Such connections will automatically become reset if an attempt is made to send data in either direction. However, half-open connections are expected to be unusual, and the recovery procedure is mildly involved.

If at site A the connection no longer exists, then an attempt by the



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 111
Keysight, EX-1045
Page 111

user at site B to send any data on it will result in the site B TCP receiving a reset control message. Such a message indicates to the site B TCP that something is wrong, and it is expected to abort the connection.

Assume that two user processes A and B are communicating with one another when a crash occurs causing loss of memory to A's TCP. Depending on the operating system supporting A's TCP, it is likely that some error recovery mechanism exists. When the TCP is up again, A is likely to start again from the beginning or from a recovery point. As a result, A will probably try to OPEN the connection again or try to SEND on the connection it believes open. In the latter case, it receives the error message "connection not open" from the local (A's) TCP. In an attempt to establish the connection, A's TCP will send a segment containing SYN. This scenario leads to the example shown in figure 10. After TCP A crashes, the user attempts to re-open the connection. TCP B, in the meantime, thinks the connection is open.

```
    TCP A                                                TCP B

1.  (CRASH)                                  (send 300,receive 100)

2.  CLOSED                                               ESTABLISHED

3.  SYN-SENT --> <SEQ=400><CTL=SYN>              --> (??)

4.  (!!)     <-- <SEQ=300><ACK=100><CTL=ACK>     <-- ESTABLISHED

5.  SYN-SENT --> <SEQ=100><CTL=RST>              --> (Abort!!)

6.  SYN-SENT                                             CLOSED

7.  SYN-SENT --> <SEQ=400><CTL=SYN>              -->
```

Half-Open Connection Discovery

Figure 10.

When the SYN arrives at line 3, TCP B, being in a synchronized state, and the incoming segment outside the window, responds with an acknowledgment indicating what sequence it next expects to hear (ACK 100). TCP A sees that this segment does not acknowledge anything it sent and, being unsynchronized, sends a reset (RST) because it has detected a half-open connection. TCP B aborts at line 5. TCP A will



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 112
Keysight, EX-1045
Page 112

continue to try to establish the connection; the problem is now reduced to the basic 3-way handshake of figure 7.

An interesting alternative case occurs when TCP A crashes and TCP B tries to send data on what it thinks is a synchronized connection. This is illustrated in figure 11. In this case, the data arriving at TCP A from TCP B (line 2) is unacceptable because no such connection exists, so TCP A sends a RST. The RST is acceptable so TCP B processes it and aborts the connection.

```
      TCP A                                                TCP B

1.  (CRASH)                                  (send 300,receive 100)

2.  (??)    <-- <SEQ=300><ACK=100><DATA=10><CTL=ACK> <-- ESTABLISHED

3.          --> <SEQ=100><CTL=RST>                   --> (ABORT!!)
```

Active Side Causes Half-Open Connection Discovery

Figure 11.

In figure 12, we find the two TCPs A and B with passive connections waiting for SYN. An old duplicate arriving at TCP B (line 2) stirs B into action. A SYN-ACK is returned (line 3) and causes TCP A to generate a RST (the ACK in line 3 is not acceptable). TCP B accepts the reset and returns to its passive LISTEN state.

```
    TCP A                                          TCP B

1.  LISTEN                                         LISTEN

2.       ... <SEQ=Z><CTL=SYN>                -->  SYN-RECEIVED

3.  (??) <-- <SEQ=X><ACK=Z+1><CTL=SYN,ACK>   <--  SYN-RECEIVED

4.       --> <SEQ=Z+1><CTL=RST>              -->  (return to LISTEN!)

5.  LISTEN                                         LISTEN
```

Old Duplicate SYN Initiates a Reset on two Passive Sockets

Figure 12.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 113
Keysight, EX-1045
Page 113

A variety of other cases are possible, all of which are accounted for by the following rules for RST generation and processing.

Reset Generation

As a general rule, reset (RST) must be sent whenever a segment arrives which apparently is not intended for the current connection. A reset must not be sent if it is not clear that this is the case.

There are three groups of states:

1. If the connection does not exist (CLOSED) then a reset is sent in response to any incoming segment except another reset. In particular, SYNs addressed to a non-existent connection are rejected by this means.

   If the incoming segment has an ACK field, the reset takes its sequence number from the ACK field of the segment, otherwise the reset has sequence number zero and the ACK field is set to the sum of the sequence number and segment length of the incoming segment. The connection remains in the CLOSED state.

2. If the connection is in any non-synchronized state (LISTEN, SYN-SENT, SYN-RECEIVED), and the incoming segment acknowledges something not yet sent (the segment carries an unacceptable ACK), or if an incoming segment has a security level or compartment which does not exactly match the level and compartment requested for the connection, a reset is sent.

   If our SYN has not been acknowledged and the precedence level of the incoming segment is higher than the precedence level requested then either raise the local precedence level (if allowed by the user and the system) or send a reset; or if the precedence level of the incoming segment is lower than the precedence level requested then continue as if the precedence matched exactly (if the remote TCP cannot raise the precedence level to match ours this will be detected in the next segment it sends, and the connection will be terminated then). If our SYN has been acknowledged (perhaps in this incoming segment) the precedence level of the incoming segment must match the local precedence level exactly, if it does not a reset must be sent.

   If the incoming segment has an ACK field, the reset takes its sequence number from the ACK field of the segment, otherwise the reset has sequence number zero and the ACK field is set to the sum of the sequence number and segment length of the incoming segment. The connection remains in the same state.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 114
Keysight, EX-1045
Page 114

3. If the connection is in a synchronized state (ESTABLISHED, FIN-WAIT-1, FIN-WAIT-2, CLOSE-WAIT, CLOSING, LAST-ACK, TIME-WAIT), any unacceptable segment (out of window sequence number or unacceptible acknowledgment number) must elicit only an empty acknowledgment segment containing the current send-sequence number and an acknowledgment indicating the next sequence number expected to be received, and the connection remains in the same state.

If an incoming segment has a security level, or compartment, or precedence which does not exactly match the level, and compartment, and precedence requested for the connection,a reset is sent and connection goes to the CLOSED state. The reset takes its sequence number from the ACK field of the incoming segment.

Reset Processing

In all states except SYN-SENT, all reset (RST) segments are validated by checking their SEQ-fields. A reset is valid if its sequence number is in the window. In the SYN-SENT state (a RST received in response to an initial SYN), the RST is acceptable if the ACK field acknowledges the SYN.

The receiver of a RST first validates it, then changes state. If the receiver was in the LISTEN state, it ignores it. If the receiver was in SYN-RECEIVED state and had previously been in the LISTEN state, then the receiver returns to the LISTEN state, otherwise the receiver aborts the connection and goes to the CLOSED state. If the receiver was in any other state, it aborts the connection and advises the user and goes to the CLOSED state.

### 3.5. Closing a Connection

CLOSE is an operation meaning "I have no more data to send." The notion of closing a full-duplex connection is subject to ambiguous interpretation, of course, since it may not be obvious how to treat the receiving side of the connection. We have chosen to treat CLOSE in a simplex fashion. The user who CLOSEs may continue to RECEIVE until he is told that the other side has CLOSED also. Thus, a program could initiate several SENDs followed by a CLOSE, and then continue to RECEIVE until signaled that a RECEIVE failed because the other side has CLOSED. We assume that the TCP will signal a user, even if no RECEIVEs are outstanding, that the other side has closed, so the user can terminate his side gracefully. A TCP will reliably deliver all buffers SENT before the connection was CLOSED so a user who expects no data in return need only wait to hear the connection was CLOSED successfully to know that all his data was received at the destination TCP. Users must keep reading connections they close for sending until the TCP says no more data.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 115
Keysight, EX-1045
Page 115

There are essentially three cases:

1) The user initiates by telling the TCP to CLOSE the connection

2) The remote TCP initiates by sending a FIN control signal

3) Both users CLOSE simultaneously

Case 1: Local user initiates the close

In this case, a FIN segment can be constructed and placed on the outgoing segment queue. No further SENDs from the user will be accepted by the TCP, and it enters the FIN-WAIT-1 state. RECEIVEs are allowed in this state. All segments preceding and including FIN will be retransmitted until acknowledged. When the other TCP has both acknowledged the FIN and sent a FIN of its own, the first TCP can ACK this FIN. Note that a TCP receiving a FIN will ACK but not send its own FIN until its user has CLOSED the connection also.

Case 2: TCP receives a FIN from the network

If an unsolicited FIN arrives from the network, the receiving TCP can ACK it and tell the user that the connection is closing. The user will respond with a CLOSE, upon which the TCP can send a FIN to the other TCP after sending any remaining data. The TCP then waits until its own FIN is acknowledged whereupon it deletes the connection. If an ACK is not forthcoming, after the user timeout the connection is aborted and the user is told.

Case 3: both users close simultaneously

A simultaneous CLOSE by users at both ends of a connection causes FIN segments to be exchanged. When all segments preceding the FINs have been processed and acknowledged, each TCP can ACK the FIN it has received. Both will, upon receiving these ACKs, delete the connection.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 116
Keysight, EX-1045
Page 116

```
    TCP A                                                TCP B

1.  ESTABLISHED                                          ESTABLISHED

2.  (Close)
    FIN-WAIT-1  --> <SEQ=100><ACK=300><CTL=FIN,ACK>  --> CLOSE-WAIT

3.  FIN-WAIT-2  <-- <SEQ=300><ACK=101><CTL=ACK>      <-- CLOSE-WAIT

4.                                                       (Close)
    TIME-WAIT   <-- <SEQ=300><ACK=101><CTL=FIN,ACK>  <-- LAST-ACK

5.  TIME-WAIT   --> <SEQ=101><ACK=301><CTL=ACK>      --> CLOSED

6.  (2 MSL)
    CLOSED
```

Normal Close Sequence

Figure 13.

```
    TCP A                                                TCP B

1.  ESTABLISHED                                          ESTABLISHED

2.  (Close)                                              (Close)
    FIN-WAIT-1  --> <SEQ=100><ACK=300><CTL=FIN,ACK>  ... FIN-WAIT-1
                <-- <SEQ=300><ACK=100><CTL=FIN,ACK>  <--
                ... <SEQ=100><ACK=300><CTL=FIN,ACK>  -->

3.  CLOSING     --> <SEQ=101><ACK=301><CTL=ACK>      ... CLOSING
                <-- <SEQ=301><ACK=101><CTL=ACK>      <--
                ... <SEQ=101><ACK=301><CTL=ACK>      -->

4.  TIME-WAIT                                            TIME-WAIT
    (2 MSL)                                              (2 MSL)
    CLOSED                                               CLOSED
```

Simultaneous Close Sequence

Figure 14.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 117
Keysight, EX-1045
Page 117

### 3.6. Precedence and Security

The intent is that connection be allowed only between ports operating with exactly the same security and compartment values and at the higher of the precedence level requested by the two ports.

The precedence and security parameters used in TCP are exactly those defined in the Internet Protocol (IP) [2]. Throughout this TCP specification the term "security/compartment" is intended to indicate the security parameters used in IP including security, compartment, user group, and handling restriction.

A connection attempt with mismatched security/compartment values or a lower precedence value must be rejected by sending a reset. Rejecting a connection due to too low a precedence only occurs after an acknowledgment of the SYN has been received.

Note that TCP modules which operate only at the default value of precedence will still have to check the precedence of incoming segments and possibly raise the precedence level they use on the connection.

The security paramaters may be used even in a non-secure environment (the values would indicate unclassified data), thus hosts in non-secure environments must be prepared to receive the security parameters, though they need not send them.

### 3.7. Data Communication

Once the connection is established data is communicated by the exchange of segments. Because segments may be lost due to errors (checksum test failure), or network congestion, TCP uses retransmission (after a timeout) to ensure delivery of every segment. Duplicate segments may arrive due to network or TCP retransmission. As discussed in the section on sequence numbers the TCP performs certain tests on the sequence and acknowledgment numbers in the segments to verify their acceptability.

The sender of data keeps track of the next sequence number to use in the variable SND.NXT. The receiver of data keeps track of the next sequence number to expect in the variable RCV.NXT. The sender of data keeps track of the oldest unacknowledged sequence number in the variable SND.UNA. If the data flow is momentarily idle and all data sent has been acknowledged then the three variables will be equal.

When the sender creates a segment and transmits it the sender advances SND.NXT. When the receiver accepts a segment it advances RCV.NXT and sends an acknowledgment. When the data sender receives an



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 118
Keysight, EX-1045
Page 118

acknowledgment it advances SND.UNA. The extent to which the values of these variables differ is a measure of the delay in the communication. The amount by which the variables are advanced is the length of the data in the segment. Note that once in the ESTABLISHED state all segments must carry current acknowledgment information.

The CLOSE user call implies a push function, as does the FIN control flag in an incoming segment.

Retransmission Timeout

Because of the variability of the networks that compose an internetwork system and the wide range of uses of TCP connections the retransmission timeout must be dynamically determined. One procedure for determining a retransmission time out is given here as an illustration.

An Example Retransmission Timeout Procedure

Measure the elapsed time between sending a data octet with a particular sequence number and receiving an acknowledgment that covers that sequence number (segments sent do not have to match segments received). This measured elapsed time is the Round Trip Time (RTT). Next compute a Smoothed Round Trip Time (SRTT) as:

```
SRTT = ( ALPHA * SRTT ) + ((1-ALPHA) * RTT)
```

and based on this, compute the retransmission timeout (RTO) as:

```
RTO = min[UBOUND,max[LBOUND,(BETA*SRTT)]]
```

where UBOUND is an upper bound on the timeout (e.g., 1 minute), LBOUND is a lower bound on the timeout (e.g., 1 second), ALPHA is a smoothing factor (e.g., .8 to .9), and BETA is a delay variance factor (e.g., 1.3 to 2.0).

The Communication of Urgent Information

The objective of the TCP urgent mechanism is to allow the sending user to stimulate the receiving user to accept some urgent data and to permit the receiving TCP to indicate to the receiving user when all the currently known urgent data has been received by the user.

This mechanism permits a point in the data stream to be designated as the end of urgent information. Whenever this point is in advance of the receive sequence number (RCV.NXT) at the receiving TCP, that TCP must tell the user to go into "urgent mode"; when the receive sequence number catches up to the urgent pointer, the TCP must tell user to go



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 119
Keysight, EX-1045
Page 119

into "normal mode". If the urgent pointer is updated while the user is in "urgent mode", the update will be invisible to the user.

The method employs a urgent field which is carried in all segments transmitted. The URG control flag indicates that the urgent field is meaningful and must be added to the segment sequence number to yield the urgent pointer. The absence of this flag indicates that there is no urgent data outstanding.

To send an urgent indication the user must also send at least one data octet. If the sending user also indicates a push, timely delivery of the urgent information to the destination process is enhanced.

Managing the Window

The window sent in each segment indicates the range of sequence numbers the sender of the window (the data receiver) is currently prepared to accept. There is an assumption that this is related to the currently available data buffer space available for this connection.

Indicating a large window encourages transmissions. If more data arrives than can be accepted, it will be discarded. This will result in excessive retransmissions, adding unnecessarily to the load on the network and the TCPs. Indicating a small window may restrict the transmission of data to the point of introducing a round trip delay between each new segment transmitted.

The mechanisms provided allow a TCP to advertise a large window and to subsequently advertise a much smaller window without having accepted that much data. This, so called "shrinking the window," is strongly discouraged. The robustness principle dictates that TCPs will not shrink the window themselves, but will be prepared for such behavior on the part of other TCPs.

The sending TCP must be prepared to accept from the user and send at least one octet of new data even if the send window is zero. The sending TCP must regularly retransmit to the receiving TCP even when the window is zero. Two minutes is recommended for the retransmission interval when the window is zero. This retransmission is essential to guarantee that when either TCP has a zero window the re-opening of the window will be reliably reported to the other.

When the receiving TCP has a zero window and a segment arrives it must still send an acknowledgment showing its next expected sequence number and current window (zero).

The sending TCP packages the data to be transmitted into segments



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 120
Keysight, EX-1045
Page 120

which fit the current window, and may repackage segments on the retransmission queue. Such repackaging is not required, but may be helpful.

In a connection with a one-way data flow, the window information will be carried in acknowledgment segments that all have the same sequence number so there will be no way to reorder them if they arrive out of order. This is not a serious problem, but it will allow the window information to be on occasion temporarily based on old reports from the data receiver. A refinement to avoid this problem is to act on the window information from segments that carry the highest acknowledgment number (that is segments with acknowledgment number equal or greater than the highest previously received).

The window management procedure has significant influence on the communication performance. The following comments are suggestions to implementers.

Window Management Suggestions

Allocating a very small window causes data to be transmitted in many small segments when better performance is achieved using fewer large segments.

One suggestion for avoiding small windows is for the receiver to defer updating a window until the additional allocation is at least X percent of the maximum allocation possible for the connection (where X might be 20 to 40).

Another suggestion is for the sender to avoid sending small segments by waiting until the window is large enough before sending data. If the the user signals a push function then the data must be sent even if it is a small segment.

Note that the acknowledgments should not be delayed or unnecessary retransmissions will result. One strategy would be to send an acknowledgment when a small segment arrives (with out updating the window information), and then to send another acknowledgment with new window information when the window is larger.

The segment sent to probe a zero window may also begin a break up of transmitted data into smaller and smaller segments. If a segment containing a single data octet sent to probe a zero window is accepted, it consumes one octet of the window now available. If the sending TCP simply sends as much as it can whenever the window is non zero, the transmitted data will be broken into alternating big and small segments. As time goes on, occasional pauses in the receiver making window allocation available will



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 121
Keysight, EX-1045
Page 121

result in breaking the big segments into a small and not quite so big pair. And after a while the data transmission will be in mostly small segments.

The suggestion here is that the TCP implementations need to actively attempt to combine small window allocations into larger windows, since the mechanisms for managing the window tend to lead to many small windows in the simplest minded implementations.

### 3.8. Interfaces

There are of course two interfaces of concern: the user/TCP interface and the TCP/lower-level interface. We have a fairly elaborate model of the user/TCP interface, but the interface to the lower level protocol module is left unspecified here, since it will be specified in detail by the specification of the lowel level protocol. For the case that the lower level is IP we note some of the parameter values that TCPs might use.

User/TCP Interface

The following functional description of user commands to the TCP is, at best, fictional, since every operating system will have different facilities. Consequently, we must warn readers that different TCP implementations may have different user interfaces. However, all TCPs must provide a certain minimum set of services to guarantee that all TCP implementations can support the same protocol hierarchy. This section specifies the functional interfaces required of all TCP implementations.

TCP User Commands

The following sections functionally characterize a USER/TCP interface. The notation used is similar to most procedure or function calls in high level languages, but this usage is not meant to rule out trap type service calls (e.g., SVCs, UUOs, EMTs).

The user commands described below specify the basic functions the TCP must perform to support interprocess communication. Individual implementations must define their own exact format, and may provide combinations or subsets of the basic functions in single calls. In particular, some implementations may wish to automatically OPEN a connection on the first SEND or RECEIVE issued by the user for a given connection.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 122
Keysight, EX-1045
Page 122

In providing interprocess communication facilities, the TCP must not only accept commands, but must also return information to the processes it serves. The latter consists of:

(a) general information about a connection (e.g., interrupts, remote close, binding of unspecified foreign socket).

(b) replies to specific user commands indicating success or various types of failure.

Open

Format: OPEN (local port, foreign socket, active/passive [, timeout] [, precedence] [, security/compartment] [, options]) -> local connection name

We assume that the local TCP is aware of the identity of the processes it serves and will check the authority of the process to use the connection specified. Depending upon the implementation of the TCP, the local network and TCP identifiers for the source address will either be supplied by the TCP or the lower level protocol (e.g., IP). These considerations are the result of concern about security, to the extent that no TCP be able to masquerade as another one, and so on. Similarly, no process can masquerade as another without the collusion of the TCP.

If the active/passive flag is set to passive, then this is a call to LISTEN for an incoming connection. A passive open may have either a fully specified foreign socket to wait for a particular connection or an unspecified foreign socket to wait for any call. A fully specified passive call can be made active by the subsequent execution of a SEND.

A transmission control block (TCB) is created and partially filled in with data from the OPEN command parameters.

On an active OPEN command, the TCP will begin the procedure to synchronize (i.e., establish) the connection at once.

The timeout, if present, permits the caller to set up a timeout for all data submitted to TCP. If data is not successfully delivered to the destination within the timeout period, the TCP will abort the connection. The present global default is five minutes.

The TCP or some component of the operating system will verify the users authority to open a connection with the specified



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 123
Keysight, EX-1045
Page 123

precedence or security/compartment. The absence of precedence or security/compartment specification in the OPEN call indicates the default values must be used.

TCP will accept incoming requests as matching only if the security/compartment information is exactly the same and only if the precedence is equal to or higher than the precedence requested in the OPEN call.

The precedence for the connection is the higher of the values requested in the OPEN call and received from the incoming request, and fixed at that value for the life of the connection.Implementers may want to give the user control of this precedence negotiation. For example, the user might be allowed to specify that the precedence must be exactly matched, or that any attempt to raise the precedence be confirmed by the user.

A local connection name will be returned to the user by the TCP. The local connection name can then be used as a short hand term for the connection defined by the <local socket, foreign socket> pair.

Send

Format: SEND (local connection name, buffer address, byte count, PUSH flag, URGENT flag [,timeout])

This call causes the data contained in the indicated user buffer to be sent on the indicated connection. If the connection has not been opened, the SEND is considered an error. Some implementations may allow users to SEND first; in which case, an automatic OPEN would be done. If the calling process is not authorized to use this connection, an error is returned.

If the PUSH flag is set, the data must be transmitted promptly to the receiver, and the PUSH bit will be set in the last TCP segment created from the buffer. If the PUSH flag is not set, the data may be combined with data from subsequent SENDs for transmission efficiency.

If the URGENT flag is set, segments sent to the destination TCP will have the urgent pointer set. The receiving TCP will signal the urgent condition to the receiving process if the urgent pointer indicates that data preceding the urgent pointer has not been consumed by the receiving process. The purpose of urgent is to stimulate the receiver to process the urgent data and to indicate to the receiver when all the currently known urgent



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 124
Keysight, EX-1045
Page 124

data has been received. The number of times the sending user's TCP signals urgent will not necessarily be equal to the number of times the receiving user will be notified of the presence of urgent data.

If no foreign socket was specified in the OPEN, but the connection is established (e.g., because a LISTENing connection has become specific due to a foreign segment arriving for the local socket), then the designated buffer is sent to the implied foreign socket. Users who make use of OPEN with an unspecified foreign socket can make use of SEND without ever explicitly knowing the foreign socket address.

However, if a SEND is attempted before the foreign socket becomes specified, an error will be returned. Users can use the STATUS call to determine the status of the connection. In some implementations the TCP may notify the user when an unspecified socket is bound.

If a timeout is specified, the current user timeout for this connection is changed to the new one.

In the simplest implementation, SEND would not return control to the sending process until either the transmission was complete or the timeout had been exceeded. However, this simple method is both subject to deadlocks (for example, both sides of the connection might try to do SENDs before doing any RECEIVEs) and offers poor performance, so it is not recommended. A more sophisticated implementation would return immediately to allow the process to run concurrently with network I/O, and, furthermore, to allow multiple SENDs to be in progress. Multiple SENDs are served in first come, first served order, so the TCP will queue those it cannot service immediately.

We have implicitly assumed an asynchronous user interface in which a SEND later elicits some kind of SIGNAL or pseudo-interrupt from the serving TCP. An alternative is to return a response immediately. For instance, SENDs might return immediate local acknowledgment, even if the segment sent had not been acknowledged by the distant TCP. We could optimistically assume eventual success. If we are wrong, the connection will close anyway due to the timeout. In implementations of this kind (synchronous), there will still be some asynchronous signals, but these will deal with the connection itself, and not with specific segments or buffers.

In order for the process to distinguish among error or success indications for different SENDs, it might be appropriate for the



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 125
Keysight, EX-1045
Page 125

buffer address to be returned along with the coded response to the SEND request. TCP-to-user signals are discussed below, indicating the information which should be returned to the calling process.

Receive

Format: RECEIVE (local connection name, buffer address, byte count) -> byte count, urgent flag, push flag

This command allocates a receiving buffer associated with the specified connection. If no OPEN precedes this command or the calling process is not authorized to use this connection, an error is returned.

In the simplest implementation, control would not return to the calling program until either the buffer was filled, or some error occurred, but this scheme is highly subject to deadlocks. A more sophisticated implementation would permit several RECEIVEs to be outstanding at once. These would be filled as segments arrive. This strategy permits increased throughput at the cost of a more elaborate scheme (possibly asynchronous) to notify the calling program that a PUSH has been seen or a buffer filled.

If enough data arrive to fill the buffer before a PUSH is seen, the PUSH flag will not be set in the response to the RECEIVE. The buffer will be filled with as much data as it can hold. If a PUSH is seen before the buffer is filled the buffer will be returned partially filled and PUSH indicated.

If there is urgent data the user will have been informed as soon as it arrived via a TCP-to-user signal. The receiving user should thus be in "urgent mode". If the URGENT flag is on, additional urgent data remains. If the URGENT flag is off, this call to RECEIVE has returned all the urgent data, and the user may now leave "urgent mode". Note that data following the urgent pointer (non-urgent data) cannot be delivered to the user in the same buffer with preceeding urgent data unless the boundary is clearly marked for the user.

To distinguish among several outstanding RECEIVEs and to take care of the case that a buffer is not completely filled, the return code is accompanied by both a buffer pointer and a byte count indicating the actual length of the data received.

Alternative implementations of RECEIVE might have the TCP



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 126
Keysight, EX-1045
Page 126

allocate buffer storage, or the TCP might share a ring buffer with the user.

Close

Format: CLOSE (local connection name)

This command causes the connection specified to be closed. If the connection is not open or the calling process is not authorized to use this connection, an error is returned. Closing connections is intended to be a graceful operation in the sense that outstanding SENDs will be transmitted (and retransmitted), as flow control permits, until all have been serviced. Thus, it should be acceptable to make several SEND calls, followed by a CLOSE, and expect all the data to be sent to the destination. It should also be clear that users should continue to RECEIVE on CLOSING connections, since the other side may be trying to transmit the last of its data. Thus, CLOSE means "I have no more to send" but does not mean "I will not receive any more." It may happen (if the user level protocol is not well thought out) that the closing side is unable to get rid of all its data before timing out. In this event, CLOSE turns into ABORT, and the closing TCP gives up.

The user may CLOSE the connection at any time on his own initiative, or in response to various prompts from the TCP (e.g., remote close executed, transmission timeout exceeded, destination inaccessible).

Because closing a connection requires communication with the foreign TCP, connections may remain in the closing state for a short time. Attempts to reopen the connection before the TCP replies to the CLOSE command will result in error responses.

Close also implies push function.

Status

Format: STATUS (local connection name) -> status data

This is an implementation dependent user command and could be excluded without adverse effect. Information returned would typically come from the TCB associated with the connection.

This command returns a data block containing the following information:

local socket,



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 127
Keysight, EX-1045
Page 127

foreign socket,
local connection name,
receive window,
send window,
connection state,
number of buffers awaiting acknowledgment,
number of buffers pending receipt,
urgent state,
precedence,
security/compartment,
and transmission timeout.

Depending on the state of the connection, or on the implementation itself, some of this information may not be available or meaningful. If the calling process is not authorized to use this connection, an error is returned. This prevents unauthorized processes from gaining information about a connection.

Abort

Format: ABORT (local connection name)

This command causes all pending SENDs and RECEIVES to be aborted, the TCB to be removed, and a special RESET message to be sent to the TCP on the other side of the connection. Depending on the implementation, users may receive abort indications for each outstanding SEND or RECEIVE, or may simply receive an ABORT-acknowledgment.

TCP-to-User Messages

It is assumed that the operating system environment provides a means for the TCP to asynchronously signal the user program. When the TCP does signal a user program, certain information is passed to the user. Often in the specification the information will be an error message. In other cases there will be information relating to the completion of processing a SEND or RECEIVE or other user call.

The following information is provided:

| | |
|---|---|
| Local Connection Name | Always |
| Response String | Always |
| Buffer Address | Send & Receive |
| Byte count (counts bytes received) | Receive |
| Push flag | Receive |
| Urgent flag | Receive |



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 128
Keysight, EX-1045
Page 128

TCP/Lower-Level Interface

The TCP calls on a lower level protocol module to actually send and receive information over a network. One case is that of the ARPA internetwork system where the lower level module is the Internet Protocol (IP) [2].

If the lower level protocol is IP it provides arguments for a type of service and for a time to live. TCP uses the following settings for these parameters:

Type of Service = Precedence: routine, Delay: normal, Throughput: normal, Reliability: normal; or 00000000.

Time to Live = one minute, or 00111100.

Note that the assumed maximum segment lifetime is two minutes. Here we explicitly ask that a segment be destroyed if it cannot be delivered by the internet system within one minute.

If the lower level is IP (or other protocol that provides this feature) and source routing is used, the interface must allow the route information to be communicated. This is especially important so that the source and destination addresses used in the TCP checksum be the originating source and ultimate destination. It is also important to preserve the return route to answer connection requests.

Any lower level protocol will have to provide the source address, destination address, and protocol fields, and some way to determine the "TCP length", both to provide the functional equivlent service of IP and to be used in the TCP checksum.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 129
Keysight, EX-1045
Page 129

### 3.9. Event Processing

The processing depicted in this section is an example of one possible implementation. Other implementations may have slightly different processing sequences, but they should differ from those in this section only in detail, not in substance.

The activity of the TCP can be characterized as responding to events. The events that occur can be cast into three categories: user calls, arriving segments, and timeouts. This section describes the processing the TCP does in response to each of the events. In many cases the processing required depends on the state of the connection.

Events that occur:

User Calls

OPEN
SEND
RECEIVE
CLOSE
ABORT
STATUS

Arriving Segments

SEGMENT ARRIVES

Timeouts

USER TIMEOUT
RETRANSMISSION TIMEOUT
TIME-WAIT TIMEOUT

The model of the TCP/user interface is that user commands receive an immediate return and possibly a delayed response via an event or pseudo interrupt. In the following descriptions, the term "signal" means cause a delayed response.

Error responses are given as character strings. For example, user commands referencing connections that do not exist receive "error: connection not open".

Please note in the following that all arithmetic on sequence numbers, acknowledgment numbers, windows, et cetera, is modulo 2**32 the size of the sequence number space. Also note that "=<" means less than or equal to (modulo 2**32).



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 130
Keysight, EX-1045
Page 130

A natural way to think about processing incoming segments is to imagine that they are first tested for proper sequence number (i.e., that their contents lie in the range of the expected "receive window" in the sequence number space) and then that they are generally queued and processed in sequence number order.

When a segment overlaps other already received segments we reconstruct the segment to contain just the new data, and adjust the header fields to be consistent.

Note that if no state change is mentioned the TCP stays in the same state.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 131
Keysight, EX-1045
Page 131

OPEN Call

CLOSED STATE (i.e., TCB does not exist)

Create a new transmission control block (TCB) to hold connection state information. Fill in local socket identifier, foreign socket, precedence, security/compartment, and user timeout information. Note that some parts of the foreign socket may be unspecified in a passive OPEN and are to be filled in by the parameters of the incoming SYN segment. Verify the security and precedence requested are allowed for this user, if not return "error: precedence not allowed" or "error: security/compartment not allowed." If passive enter the LISTEN state and return. If active and the foreign socket is unspecified, return "error: foreign socket unspecified"; if active and the foreign socket is specified, issue a SYN segment. An initial send sequence number (ISS) is selected. A SYN segment of the form <SEQ=ISS><CTL=SYN> is sent. Set SND.UNA to ISS, SND.NXT to ISS+1, enter SYN-SENT state, and return.

If the caller does not have access to the local socket specified, return "error: connection illegal for this process". If there is no room to create a new connection, return "error: insufficient resources".

LISTEN STATE

If active and the foreign socket is specified, then change the connection from passive to active, select an ISS. Send a SYN segment, set SND.UNA to ISS, SND.NXT to ISS+1. Enter SYN-SENT state. Data associated with SEND may be sent with SYN segment or queued for transmission after entering ESTABLISHED state. The urgent bit if requested in the command must be sent with the data segments sent as a result of this command. If there is no room to queue the request, respond with "error: insufficient resources". If Foreign socket was not specified, then return "error: foreign socket unspecified".



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 132
Keysight, EX-1045
Page 132

OPEN Call

SYN-SENT STATE
SYN-RECEIVED STATE
ESTABLISHED STATE
FIN-WAIT-1 STATE
FIN-WAIT-2 STATE
CLOSE-WAIT STATE
CLOSING STATE
LAST-ACK STATE
TIME-WAIT STATE

Return "error: connection already exists".



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 133
Keysight, EX-1045
Page 133

SEND Call

CLOSED STATE (i.e., TCB does not exist)

If the user does not have access to such a connection, then return "error: connection illegal for this process".

Otherwise, return "error: connection does not exist".

LISTEN STATE

If the foreign socket is specified, then change the connection from passive to active, select an ISS. Send a SYN segment, set SND.UNA to ISS, SND.NXT to ISS+1. Enter SYN-SENT state. Data associated with SEND may be sent with SYN segment or queued for transmission after entering ESTABLISHED state. The urgent bit if requested in the command must be sent with the data segments sent as a result of this command. If there is no room to queue the request, respond with "error: insufficient resources". If Foreign socket was not specified, then return "error: foreign socket unspecified".

SYN-SENT STATE
SYN-RECEIVED STATE

Queue the data for transmission after entering ESTABLISHED state. If no space to queue, respond with "error: insufficient resources".

ESTABLISHED STATE
CLOSE-WAIT STATE

Segmentize the buffer and send it with a piggybacked acknowledgment (acknowledgment value = RCV.NXT). If there is insufficient space to remember this buffer, simply return "error: insufficient resources".

If the urgent flag is set, then SND.UP <- SND.NXT-1 and set the urgent pointer in the outgoing segments.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 134
Keysight, EX-1045
Page 134

SEND Call

FIN-WAIT-1 STATE
FIN-WAIT-2 STATE
CLOSING STATE
LAST-ACK STATE
TIME-WAIT STATE

Return "error: connection closing" and do not service request.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 135
Keysight, EX-1045
Page 135

RECEIVE Call

CLOSED STATE (i.e., TCB does not exist)

If the user does not have access to such a connection, return "error: connection illegal for this process".

Otherwise return "error: connection does not exist".

LISTEN STATE
SYN-SENT STATE
SYN-RECEIVED STATE

Queue for processing after entering ESTABLISHED state. If there is no room to queue this request, respond with "error: insufficient resources".

ESTABLISHED STATE
FIN-WAIT-1 STATE
FIN-WAIT-2 STATE

If insufficient incoming segments are queued to satisfy the request, queue the request. If there is no queue space to remember the RECEIVE, respond with "error: insufficient resources".

Reassemble queued incoming segments into receive buffer and return to user. Mark "push seen" (PUSH) if this is the case.

If RCV.UP is in advance of the data currently being passed to the user notify the user of the presence of urgent data.

When the TCP takes responsibility for delivering data to the user that fact must be communicated to the sender via an acknowledgment. The formation of such an acknowledgment is described below in the discussion of processing an incoming segment.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 136
Keysight, EX-1045
Page 136

RECEIVE Call

CLOSE-WAIT STATE

Since the remote side has already sent FIN, RECEIVEs must be satisfied by text already on hand, but not yet delivered to the user. If no text is awaiting delivery, the RECEIVE will get a "error: connection closing" response. Otherwise, any remaining text can be used to satisfy the RECEIVE.

CLOSING STATE
LAST-ACK STATE
TIME-WAIT STATE

Return "error: connection closing".



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 137
Keysight, EX-1045
Page 137

CLOSE Call

CLOSED STATE (i.e., TCB does not exist)

If the user does not have access to such a connection, return "error: connection illegal for this process".

Otherwise, return "error: connection does not exist".

LISTEN STATE

Any outstanding RECEIVEs are returned with "error: closing" responses. Delete TCB, enter CLOSED state, and return.

SYN-SENT STATE

Delete the TCB and return "error: closing" responses to any queued SENDs, or RECEIVEs.

SYN-RECEIVED STATE

If no SENDs have been issued and there is no pending data to send, then form a FIN segment and send it, and enter FIN-WAIT-1 state; otherwise queue for processing after entering ESTABLISHED state.

ESTABLISHED STATE

Queue this until all preceding SENDs have been segmentized, then form a FIN segment and send it. In any case, enter FIN-WAIT-1 state.

FIN-WAIT-1 STATE
FIN-WAIT-2 STATE

Strictly speaking, this is an error and should receive a "error: connection closing" response. An "ok" response would be acceptable, too, as long as a second FIN is not emitted (the first FIN may be retransmitted though).



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 138
Keysight, EX-1045
Page 138

CLOSE Call

CLOSE-WAIT STATE

Queue this request until all preceding SENDs have been segmentized; then send a FIN segment, enter CLOSING state.

CLOSING STATE
LAST-ACK STATE
TIME-WAIT STATE

Respond with "error: connection closing".



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 139
Keysight, EX-1045
Page 139

ABORT Call

CLOSED STATE (i.e., TCB does not exist)

If the user should not have access to such a connection, return "error: connection illegal for this process".

Otherwise return "error: connection does not exist".

LISTEN STATE

Any outstanding RECEIVEs should be returned with "error: connection reset" responses. Delete TCB, enter CLOSED state, and return.

SYN-SENT STATE

All queued SENDs and RECEIVEs should be given "connection reset" notification, delete the TCB, enter CLOSED state, and return.

SYN-RECEIVED STATE
ESTABLISHED STATE
FIN-WAIT-1 STATE
FIN-WAIT-2 STATE
CLOSE-WAIT STATE

Send a reset segment:

<SEQ=SND.NXT><CTL=RST>

All queued SENDs and RECEIVEs should be given "connection reset" notification; all segments queued for transmission (except for the RST formed above) or retransmission should be flushed, delete the TCB, enter CLOSED state, and return.

CLOSING STATE
LAST-ACK STATE
TIME-WAIT STATE

Respond with "ok" and delete the TCB, enter CLOSED state, and return.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 140
Keysight, EX-1045
Page 140

STATUS Call

CLOSED STATE (i.e., TCB does not exist)

If the user should not have access to such a connection, return "error: connection illegal for this process".

Otherwise return "error: connection does not exist".

LISTEN STATE

Return "state = LISTEN", and the TCB pointer.

SYN-SENT STATE

Return "state = SYN-SENT", and the TCB pointer.

SYN-RECEIVED STATE

Return "state = SYN-RECEIVED", and the TCB pointer.

ESTABLISHED STATE

Return "state = ESTABLISHED", and the TCB pointer.

FIN-WAIT-1 STATE

Return "state = FIN-WAIT-1", and the TCB pointer.

FIN-WAIT-2 STATE

Return "state = FIN-WAIT-2", and the TCB pointer.

CLOSE-WAIT STATE

Return "state = CLOSE-WAIT", and the TCB pointer.

CLOSING STATE

Return "state = CLOSING", and the TCB pointer.

LAST-ACK STATE

Return "state = LAST-ACK", and the TCB pointer.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 141
Keysight, EX-1045
Page 141

STATUS Call

TIME-WAIT STATE

Return "state = TIME-WAIT", and the TCB pointer.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 142
Keysight, EX-1045
Page 142

SEGMENT ARRIVES

SEGMENT ARRIVES

If the state is CLOSED (i.e., TCB does not exist) then

all data in the incoming segment is discarded. An incoming segment containing a RST is discarded. An incoming segment not containing a RST causes a RST to be sent in response. The acknowledgment and sequence field values are selected to make the reset sequence acceptable to the TCP that sent the offending segment.

If the ACK bit is off, sequence number zero is used,

<SEQ=0><ACK=SEG.SEQ+SEG.LEN><CTL=RST,ACK>

If the ACK bit is on,

<SEQ=SEG.ACK><CTL=RST>

Return.

If the state is LISTEN then

first check for an RST

An incoming RST should be ignored. Return.

second check for an ACK

Any acknowledgment is bad if it arrives on a connection still in the LISTEN state. An acceptable reset segment should be formed for any arriving ACK-bearing segment. The RST should be formatted as follows:

<SEQ=SEG.ACK><CTL=RST>

Return.

third check for a SYN

If the SYN bit is set, check the security. If the security/compartment on the incoming segment does not exactly match the security/compartment in the TCB then send a reset and return.

<SEQ=SEG.ACK><CTL=RST>



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 143
Keysight, EX-1045
Page 143

If the SEG.PRC is greater than the TCB.PRC then if allowed by the user and the system set TCB.PRC<-SEG.PRC, if not allowed send a reset and return.

<SEQ=SEG.ACK><CTL=RST>

If the SEG.PRC is less than the TCB.PRC then continue.

Set RCV.NXT to SEG.SEQ+1, IRS is set to SEG.SEQ and any other control or text should be queued for processing later. ISS should be selected and a SYN segment sent of the form:

<SEQ=ISS><ACK=RCV.NXT><CTL=SYN,ACK>

SND.NXT is set to ISS+1 and SND.UNA to ISS. The connection state should be changed to SYN-RECEIVED. Note that any other incoming control or data (combined with SYN) will be processed in the SYN-RECEIVED state, but processing of SYN and ACK should not be repeated. If the listen was not fully specified (i.e., the foreign socket was not fully specified), then the unspecified fields should be filled in now.

fourth other text or control

Any other control or text-bearing segment (not containing SYN) must have an ACK and thus would be discarded by the ACK processing. An incoming RST segment could not be valid, since it could not have been sent in response to anything sent by this incarnation of the connection. So you are unlikely to get here, but if you do, drop the segment, and return.

If the state is SYN-SENT then

first check the ACK bit

If the ACK bit is set

If SEG.ACK =< ISS, or SEG.ACK > SND.NXT, send a reset (unless the RST bit is set, if so drop the segment and return)

<SEQ=SEG.ACK><CTL=RST>

and discard the segment. Return.

If SND.UNA =< SEG.ACK =< SND.NXT then the ACK is acceptable.

second check the RST bit



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 144
Keysight, EX-1045
Page 144

SEGMENT ARRIVES

If the RST bit is set

If the ACK was acceptable then signal the user "error: connection reset", drop the segment, enter CLOSED state, delete TCB, and return. Otherwise (no ACK) drop the segment and return.

third check the security and precedence

If the security/compartment in the segment does not exactly match the security/compartment in the TCB, send a reset

If there is an ACK

<SEQ=SEG.ACK><CTL=RST>

Otherwise

<SEQ=0><ACK=SEG.SEQ+SEG.LEN><CTL=RST,ACK>

If there is an ACK

The precedence in the segment must match the precedence in the TCB, if not, send a reset

<SEQ=SEG.ACK><CTL=RST>

If there is no ACK

If the precedence in the segment is higher than the precedence in the TCB then if allowed by the user and the system raise the precedence in the TCB to that in the segment, if not allowed to raise the prec then send a reset.

<SEQ=0><ACK=SEG.SEQ+SEG.LEN><CTL=RST,ACK>

If the precedence in the segment is lower than the precedence in the TCB continue.

If a reset was sent, discard the segment and return.

fourth check the SYN bit

This step should be reached only if the ACK is ok, or there is no ACK, and it the segment did not contain a RST.

If the SYN bit is on and the security/compartment and precedence



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 145
Keysight, EX-1045
Page 145

are acceptable then, RCV.NXT is set to SEG.SEQ+1, IRS is set to SEG.SEQ. SND.UNA should be advanced to equal SEG.ACK (if there is an ACK), and any segments on the retransmission queue which are thereby acknowledged should be removed.

If SND.UNA > ISS (our SYN has been ACKed), change the connection state to ESTABLISHED, form an ACK segment

<SEQ=SND.NXT><ACK=RCV.NXT><CTL=ACK>

and send it. Data or controls which were queued for transmission may be included. If there are other controls or text in the segment then continue processing at the sixth step below where the URG bit is checked, otherwise return.

Otherwise enter SYN-RECEIVED, form a SYN,ACK segment

<SEQ=ISS><ACK=RCV.NXT><CTL=SYN,ACK>

and send it. If there are other controls or text in the segment, queue them for processing after the ESTABLISHED state has been reached, return.

fifth, if neither of the SYN or RST bits is set then drop the segment and return.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 146
Keysight, EX-1045
Page 146

SEGMENT ARRIVES

Otherwise,

first check sequence number

SYN-RECEIVED STATE
ESTABLISHED STATE
FIN-WAIT-1 STATE
FIN-WAIT-2 STATE
CLOSE-WAIT STATE
CLOSING STATE
LAST-ACK STATE
TIME-WAIT STATE

Segments are processed in sequence. Initial tests on arrival are used to discard old duplicates, but further processing is done in SEG.SEQ order. If a segment's contents straddle the boundary between old and new, only the new parts should be processed.

There are four cases for the acceptability test for an incoming segment:

| Segment Length | Receive Window | Test |
|---|---|---|
| 0 | 0 | SEG.SEQ = RCV.NXT |
| 0 | >0 | RCV.NXT =< SEG.SEQ < RCV.NXT+RCV.WND |
| >0 | 0 | not acceptable |
| >0 | >0 | RCV.NXT =< SEG.SEQ < RCV.NXT+RCV.WND<br>or RCV.NXT =< SEG.SEQ+SEG.LEN-1 < RCV.NXT+RCV.WND |

If the RCV.WND is zero, no segments will be acceptable, but special allowance should be made to accept valid ACKs, URGs and RSTs.

If an incoming segment is not acceptable, an acknowledgment should be sent in reply (unless the RST bit is set, if so drop the segment and return):

<SEQ=SND.NXT><ACK=RCV.NXT><CTL=ACK>

After sending the acknowledgment, drop the unacceptable segment and return.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 147
Keysight, EX-1045
Page 147

In the following it is assumed that the segment is the idealized segment that begins at RCV.NXT and does not exceed the window. One could tailor actual segments to fit this assumption by trimming off any portions that lie outside the window (including SYN and FIN), and only processing further if the segment then begins at RCV.NXT. Segments with higher begining sequence numbers may be held for later processing.

second check the RST bit,

SYN-RECEIVED STATE

If the RST bit is set

If this connection was initiated with a passive OPEN (i.e., came from the LISTEN state), then return this connection to LISTEN state and return. The user need not be informed. If this connection was initiated with an active OPEN (i.e., came from SYN-SENT state) then the connection was refused, signal the user "connection refused". In either case, all segments on the retransmission queue should be removed. And in the active OPEN case, enter the CLOSED state and delete the TCB, and return.

ESTABLISHED
FIN-WAIT-1
FIN-WAIT-2
CLOSE-WAIT

If the RST bit is set then, any outstanding RECEIVEs and SEND should receive "reset" responses. All segment queues should be flushed. Users should also receive an unsolicited general "connection reset" signal. Enter the CLOSED state, delete the TCB, and return.

CLOSING STATE
LAST-ACK STATE
TIME-WAIT

If the RST bit is set then, enter the CLOSED state, delete the TCB, and return.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 148
Keysight, EX-1045
Page 148

SEGMENT ARRIVES

third check security and precedence

SYN-RECEIVED

If the security/compartment and precedence in the segment do not exactly match the security/compartment and precedence in the TCB then send a reset, and return.

ESTABLISHED STATE

If the security/compartment and precedence in the segment do not exactly match the security/compartment and precedence in the TCB then send a reset, any outstanding RECEIVEs and SEND should receive "reset" responses. All segment queues should be flushed. Users should also receive an unsolicited general "connection reset" signal. Enter the CLOSED state, delete the TCB, and return.

Note this check is placed following the sequence check to prevent a segment from an old connection between these ports with a different security or precedence from causing an abort of the current connection.

fourth, check the SYN bit,

SYN-RECEIVED
ESTABLISHED STATE
FIN-WAIT STATE-1
FIN-WAIT STATE-2
CLOSE-WAIT STATE
CLOSING STATE
LAST-ACK STATE
TIME-WAIT STATE

If the SYN is in the window it is an error, send a reset, any outstanding RECEIVEs and SEND should receive "reset" responses, all segment queues should be flushed, the user should also receive an unsolicited general "connection reset" signal, enter the CLOSED state, delete the TCB, and return.

If the SYN is not in the window this step would not be reached and an ack would have been sent in the first step (sequence number check).



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 149
Keysight, EX-1045
Page 149

fifth check the ACK field,

if the ACK bit is off drop the segment and return

if the ACK bit is on

SYN-RECEIVED STATE

If SND.UNA =< SEG.ACK =< SND.NXT then enter ESTABLISHED state and continue processing.

If the segment acknowledgment is not acceptable, form a reset segment,

<SEQ=SEG.ACK><CTL=RST>

and send it.

ESTABLISHED STATE

If SND.UNA < SEG.ACK =< SND.NXT then, set SND.UNA <- SEG.ACK. Any segments on the retransmission queue which are thereby entirely acknowledged are removed. Users should receive positive acknowledgments for buffers which have been SENT and fully acknowledged (i.e., SEND buffer should be returned with "ok" response). If the ACK is a duplicate (SEG.ACK < SND.UNA), it can be ignored. If the ACK acks something not yet sent (SEG.ACK > SND.NXT) then send an ACK, drop the segment, and return.

If SND.UNA < SEG.ACK =< SND.NXT, the send window should be updated. If (SND.WL1 < SEG.SEQ or (SND.WL1 = SEG.SEQ and SND.WL2 =< SEG.ACK)), set SND.WND <- SEG.WND, set SND.WL1 <- SEG.SEQ, and set SND.WL2 <- SEG.ACK.

Note that SND.WND is an offset from SND.UNA, that SND.WL1 records the sequence number of the last segment used to update SND.WND, and that SND.WL2 records the acknowledgment number of the last segment used to update SND.WND. The check here prevents using old segments to update the window.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 150
Keysight, EX-1045
Page 150

SEGMENT ARRIVES

FIN-WAIT-1 STATE

In addition to the processing for the ESTABLISHED state, if our FIN is now acknowledged then enter FIN-WAIT-2 and continue processing in that state.

FIN-WAIT-2 STATE

In addition to the processing for the ESTABLISHED state, if the retransmission queue is empty, the user's CLOSE can be acknowledged ("ok") but do not delete the TCB.

CLOSE-WAIT STATE

Do the same processing as for the ESTABLISHED state.

CLOSING STATE

In addition to the processing for the ESTABLISHED state, if the ACK acknowledges our FIN then enter the TIME-WAIT state, otherwise ignore the segment.

LAST-ACK STATE

The only thing that can arrive in this state is an acknowledgment of our FIN. If our FIN is now acknowledged, delete the TCB, enter the CLOSED state, and return.

TIME-WAIT STATE

The only thing that can arrive in this state is a retransmission of the remote FIN. Acknowledge it, and restart the 2 MSL timeout.

sixth, check the URG bit,

ESTABLISHED STATE
FIN-WAIT-1 STATE
FIN-WAIT-2 STATE

If the URG bit is set, RCV.UP <- max(RCV.UP,SEG.UP), and signal the user that the remote side has urgent data if the urgent pointer (RCV.UP) is in advance of the data consumed. If the user has already been signaled (or is still in the "urgent mode") for this continuous sequence of urgent data, do not signal the user again.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 151
Keysight, EX-1045
Page 151

CLOSE-WAIT STATE
CLOSING STATE
LAST-ACK STATE
TIME-WAIT

This should not occur, since a FIN has been received from the remote side. Ignore the URG.

seventh, process the segment text,

ESTABLISHED STATE
FIN-WAIT-1 STATE
FIN-WAIT-2 STATE

Once in the ESTABLISHED state, it is possible to deliver segment text to user RECEIVE buffers. Text from segments can be moved into buffers until either the buffer is full or the segment is empty. If the segment empties and carries an PUSH flag, then the user is informed, when the buffer is returned, that a PUSH has been received.

When the TCP takes responsibility for delivering the data to the user it must also acknowledge the receipt of the data.

Once the TCP takes responsibility for the data it advances RCV.NXT over the data accepted, and adjusts RCV.WND as apporopriate to the current buffer availability. The total of RCV.NXT and RCV.WND should not be reduced.

Please note the window management suggestions in section 3.7.

Send an acknowledgment of the form:

<SEQ=SND.NXT><ACK=RCV.NXT><CTL=ACK>

This acknowledgment should be piggybacked on a segment being transmitted if possible without incurring undue delay.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 152
Keysight, EX-1045
Page 152

SEGMENT ARRIVES

CLOSE-WAIT STATE
CLOSING STATE
LAST-ACK STATE
TIME-WAIT STATE

This should not occur, since a FIN has been received from the remote side. Ignore the segment text.

eighth, check the FIN bit,

Do not process the FIN if the state is CLOSED, LISTEN or SYN-SENT since the SEG.SEQ cannot be validated; drop the segment and return.

If the FIN bit is set, signal the user "connection closing" and return any pending RECEIVEs with same message, advance RCV.NXT over the FIN, and send an acknowledgment for the FIN. Note that FIN implies PUSH for any segment text not yet delivered to the user.

SYN-RECEIVED STATE
ESTABLISHED STATE

Enter the CLOSE-WAIT state.

FIN-WAIT-1 STATE

If our FIN has been ACKed (perhaps in this segment), then enter TIME-WAIT, start the time-wait timer, turn off the other timers; otherwise enter the CLOSING state.

FIN-WAIT-2 STATE

Enter the TIME-WAIT state. Start the time-wait timer, turn off the other timers.

CLOSE-WAIT STATE

Remain in the CLOSE-WAIT state.

CLOSING STATE

Remain in the CLOSING state.

LAST-ACK STATE

Remain in the LAST-ACK state.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 153
Keysight, EX-1045
Page 153

SEGMENT ARRIVES

TIME-WAIT STATE

Remain in the TIME-WAIT state. Restart the 2 MSL time-wait timeout.

and return.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 154
Keysight, EX-1045
Page 154

USER TIMEOUT

For any state if the user timeout expires, flush all queues, signal the user "error: connection aborted due to user timeout" in general and for any outstanding calls, delete the TCB, enter the CLOSED state and return.

RETRANSMISSION TIMEOUT

For any state if the retransmission timeout expires on a segment in the retransmission queue, send the segment at the front of the retransmission queue again, reinitialize the retransmission timer, and return.

TIME-WAIT TIMEOUT

If the time-wait timeout expires on a connection delete the TCB, enter the CLOSED state and return.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 155
Keysight, EX-1045
Page 155

https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 156
**Keysight, EX-1045**
**Page 156**

## GLOSSARY

1822
: BBN Report 1822, "The Specification of the Interconnection of a Host and an IMP". The specification of interface between a host and the ARPANET.

ACK
: A control bit (acknowledge) occupying no sequence space, which indicates that the acknowledgment field of this segment specifies the next sequence number the sender of this segment is expecting to receive, hence acknowledging receipt of all previous sequence numbers.

ARPANET message
: The unit of transmission between a host and an IMP in the ARPANET. The maximum size is about 1012 octets (8096 bits).

ARPANET packet
: A unit of transmission used internally in the ARPANET between IMPs. The maximum size is about 126 octets (1008 bits).

connection
: A logical communication path identified by a pair of sockets.

datagram
: A message sent in a packet switched computer communications network.

Destination Address
: The destination address, usually the network and host identifiers.

FIN
: A control bit (finis) occupying one sequence number, which indicates that the sender will send no more data or control occupying sequence space.

fragment
: A portion of a logical unit of data, in particular an internet fragment is a portion of an internet datagram.

FTP
: A file transfer protocol.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 157
Keysight, EX-1045
Page 157

header
Control information at the beginning of a message, segment, fragment, packet or block of data.

host
A computer. In particular a source or destination of messages from the point of view of the communication network.

Identification
An Internet Protocol field. This identifying value assigned by the sender aids in assembling the fragments of a datagram.

IMP
The Interface Message Processor, the packet switch of the ARPANET.

internet address
A source or destination address specific to the host level.

internet datagram
The unit of data exchanged between an internet module and the higher level protocol together with the internet header.

internet fragment
A portion of the data of an internet datagram with an internet header.

IP
Internet Protocol.

IRS
The Initial Receive Sequence number. The first sequence number used by the sender on a connection.

ISN
The Initial Sequence Number. The first sequence number used on a connection, (either ISS or IRS). Selected on a clock based procedure.

ISS
The Initial Send Sequence number. The first sequence number used by the sender on a connection.

leader
Control information at the beginning of a message or block of data. In particular, in the ARPANET, the control information on an ARPANET message at the host-IMP interface.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 158
Keysight, EX-1045
Page 158

left sequence
: This is the next sequence number to be acknowledged by the data receiving TCP (or the lowest currently unacknowledged sequence number) and is sometimes referred to as the left edge of the send window.

local packet
: The unit of transmission within a local network.

module
: An implementation, usually in software, of a protocol or other procedure.

MSL
: Maximum Segment Lifetime, the time a TCP segment can exist in the internetwork system. Arbitrarily defined to be 2 minutes.

octet
: An eight bit byte.

Options
: An Option field may contain several options, and each option may be several octets in length. The options are used primarily in testing situations; for example, to carry timestamps. Both the Internet Protocol and TCP provide for options fields.

packet
: A package of data with a header which may or may not be logically complete. More often a physical packaging than a logical packaging of data.

port
: The portion of a socket that specifies which logical input or output channel of a process is associated with the data.

process
: A program in execution. A source or destination of data from the point of view of the TCP or other host-to-host protocol.

PUSH
: A control bit occupying no sequence space, indicating that this segment contains data that must be pushed through to the receiving user.

RCV.NXT
: receive next sequence number



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 159
Keysight, EX-1045
Page 159

RCV.UP
receive urgent pointer

RCV.WND
receive window

receive next sequence number
This is the next sequence number the local TCP is expecting to receive.

receive window
This represents the sequence numbers the local (receiving) TCP is willing to receive. Thus, the local TCP considers that segments overlapping the range RCV.NXT to RCV.NXT + RCV.WND - 1 carry acceptable data or control. Segments containing sequence numbers entirely outside of this range are considered duplicates and discarded.

RST
A control bit (reset), occupying no sequence space, indicating that the receiver should delete the connection without further interaction. The receiver can determine, based on the sequence number and acknowledgment fields of the incoming segment, whether it should honor the reset command or ignore it. In no case does receipt of a segment containing RST give rise to a RST in response.

RTP
Real Time Protocol: A host-to-host protocol for communication of time critical information.

SEG.ACK
segment acknowledgment

SEG.LEN
segment length

SEG.PRC
segment precedence value

SEG.SEQ
segment sequence

SEG.UP
segment urgent pointer field



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 160
Keysight, EX-1045
Page 160

SEG.WND
segment window field

segment
A logical unit of data, in particular a TCP segment is the unit of data transfered between a pair of TCP modules.

segment acknowledgment
The sequence number in the acknowledgment field of the arriving segment.

segment length
The amount of sequence number space occupied by a segment, including any controls which occupy sequence space.

segment sequence
The number in the sequence field of the arriving segment.

send sequence
This is the next sequence number the local (sending) TCP will use on the connection. It is initially selected from an initial sequence number curve (ISN) and is incremented for each octet of data or sequenced control transmitted.

send window
This represents the sequence numbers which the remote (receiving) TCP is willing to receive. It is the value of the window field specified in segments from the remote (data receiving) TCP. The range of new sequence numbers which may be emitted by a TCP lies between SND.NXT and SND.UNA + SND.WND - 1. (Retransmissions of sequence numbers between SND.UNA and SND.NXT are expected, of course.)

SND.NXT
send sequence

SND.UNA
left sequence

SND.UP
send urgent pointer

SND.WL1
segment sequence number at last window update

SND.WL2
segment acknowledgment number at last window update



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 161
Keysight, EX-1045
Page 161

SND.WND
send window

socket
An address which specifically includes a port identifier, that is, the concatenation of an Internet Address with a TCP port.

Source Address
The source address, usually the network and host identifiers.

SYN
A control bit in the incoming segment, occupying one sequence number, used at the initiation of a connection, to indicate where the sequence numbering will start.

TCB
Transmission control block, the data structure that records the state of a connection.

TCB.PRC
The precedence of the connection.

TCP
Transmission Control Protocol: A host-to-host protocol for reliable communication in internetwork environments.

TOS
Type of Service, an Internet Protocol field.

Type of Service
An Internet Protocol field which indicates the type of service for this internet fragment.

URG
A control bit (urgent), occupying no sequence space, used to indicate that the receiving user should be notified to do urgent processing as long as there is data to be consumed with sequence numbers less than the value indicated in the urgent pointer.

urgent pointer
A control field meaningful only when the URG bit is on. This field communicates the value of the urgent pointer which indicates the data octet associated with the sending user's urgent call.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 162
Keysight, EX-1045
Page 162

## REFERENCES

[1] Cerf, V., and R. Kahn, "A Protocol for Packet Network Intercommunication", IEEE Transactions on Communications, Vol. COM-22, No. 5, pp 637-648, May 1974.

[2] Postel, J. (ed.), "Internet Protocol - DARPA Internet Program Protocol Specification", RFC 791, USC/Information Sciences Institute, September 1981.

[3] Dalal, Y. and C. Sunshine, "Connection Management in Transport Protocols", Computer Networks, Vol. 2, No. 6, pp. 454-473, December 1978.

[4] Postel, J., "Assigned Numbers", RFC 790, USC/Information Sciences Institute, September 1981.



https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 163
Keysight, EX-1045
Page 163

https://web.archive.org/web/20070202201546/https://tools.ietf.org/html/rfc793

Network Working Group
Request for Comments: 1919
Category: Informational

M. Chatel
Consultant

March 1996

# Classical versus Transparent IP Proxies

## Status of this Memo

This memo provides information for the Internet community. This memo does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

## Abstract

Many modern IP security systems (also called "firewalls" in the trade) make use of proxy technology to achieve access control. This document explains "classical" and "transparent" proxy techniques and attempts to provide rules to help determine when each proxy system may be used without causing problems.

## Table of Contents

1. Background . . . . . 2
2. Direct communication (without a proxy) . . . . . 3
2.1. Direct connection example . . . . . 3
2.2. Requirements of direct communication . . . . . 5
3. Classical application proxies . . . . . 5
3.1. Classical proxy session example . . . . . 6
3.2. Characteristics of classical proxy configurations . . . 12
3.2.1. IP addressing and routing requirements . . . . . 12
3.2.2. IP address hiding . . . . . 14
3.2.3. DNS requirements . . . . . 14
3.2.4. Software requirements . . . . . 15
3.2.5. Impact of a classical proxy on packet filtering . . . 15
3.2.6. Interconnection of conflicting IP networks . . . . . 16
4. Transparent application proxies . . . . . 19
4.1. Transparent proxy connection example . . . . . 20
4.2. Characteristics of transparent proxy configurations . . 26
4.2.1. IP addressing and routing requirements . . . . . 26
4.2.2. IP address hiding . . . . . 28
4.2.3. DNS requirements . . . . . 28
4.2.4. Software requirements . . . . . 29
4.2.5. Impact of a transparent proxy on packet filtering . . 30
4.2.6. Interconnection of conflicting IP networks . . . . . 31
5. Comparison chart of classical and transparent proxies . . 31
6. Improving transparent proxies . . . . . 32
7. Security Considerations . . . . . 34



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 165
Keysight, EX-1045
Page 165

8. Acknowledgements . . . . . 34
9. References . . . . . 35

## 1. Background

An increasing number of organizations use IP security systems to provide specific access control when crossing network security perimeters. These systems are often deployed at the network boundary between two organizations (which may be part of the same "official" entity), or between an organization's network and a large public internetwork such as the Internet.

Some people believe that IP firewalls will become commodity products. Others believe that the introduction of IPv6 and of its improved security capabilities will gradually make firewalls look like stopgap solutions, and therefore irrelevant to the computer networking scene. In any case, it is currently important to examine the impact of inserting (and removing) a firewall at a network boundary, and to verify whether specific types of firewall technologies may have different effects on typical small and large IP networks.

Current firewall designs usually rely on packet filtering, proxy technology, or a combination of both. Packet filtering (although hard to configure correctly in a security sense) is now a well documented technology whose strengths and weaknesses are reasonably understood. Proxy technology, on the other hand, has been deployed a lot but studied little. Furthermore, many recent firewall products support a capability called "transparent proxying". This type of feature has been subject to much more marketing attention than actual technical analysis by the networking community.

It must be remembered that the Internet's growth and success is strongly related to its "open" nature. An Internet which would have been segmented from the start with firewalls, packet filters, and proxies may not have become what it is today. This type of discussion is, however, outside the scope of this document, which just attempts to provide an understandable description of what are network proxies, and of what are the differences, strengths, and weaknesses of "classical" and "transparent" network proxies. Within the context of this document, a "classical" proxy is the older (some would say old-fashioned) type of proxy of the two.

Also note that in this document, the word "connection" is used for an application session that uses TCP, while the word "session" refers to an application dialog that may use UDP or TCP.



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

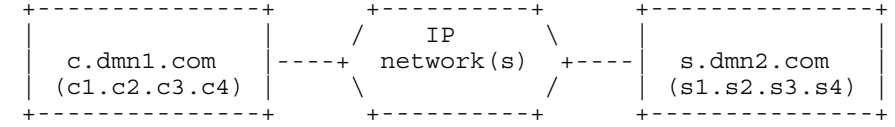
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 166
Keysight, EX-1045
Page 166

## 2. Direct communication (without a proxy)

In the "normal" Internet world, systems do not use proxies and simply use normal TCP/IP to communicate with each other. It is important (for readers who may not be familiar with this) to take a quick look at the operations involved, in order to better understand what is the exact use of a proxy.

### 2.1 Direct connection example

Let's take a familiar network session and describe some details of its operation. We will look at what happens when a user on a client system "c.dmn1.com" sets up an FTP connection to the server system "s.dmn2.com". The client system's IP address is c1.c2.c3.c4, the server's IP address is s1.s2.s3.s4.




The user starts an instance of an FTP client program on the client system "c.dmn1.com", and specifies that the target system is "s.dmn2.com". On command-line systems, the user typically types:

```
ftp s.dmn2.com
```

The client system needs to convert the server's name to an IP address (if the user directly specified the server by address, this step is not needed).

Converting the server name to an IP address requires work to be performed which ranges between two extremes:

a) the client system has this name in its hosts file, or has local DNS caching capability and successfully retrieves the name of the server system in its cache. No network activity is performed to convert the name to an IP address.

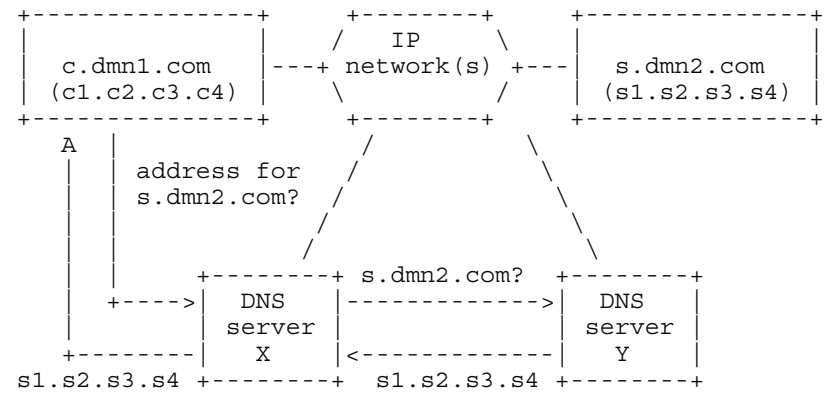
b) the client system, in combination with DNS name servers, generate DNS queries that eventually propagate close to the root of the DNS tree and back down the server's DNS branch. Eventually, a DNS server which is authoritative for the server system's domain is queried and returns the IP address associated with "s.dmn2.com" (depending on the case, it may return this to the client system directly or to an



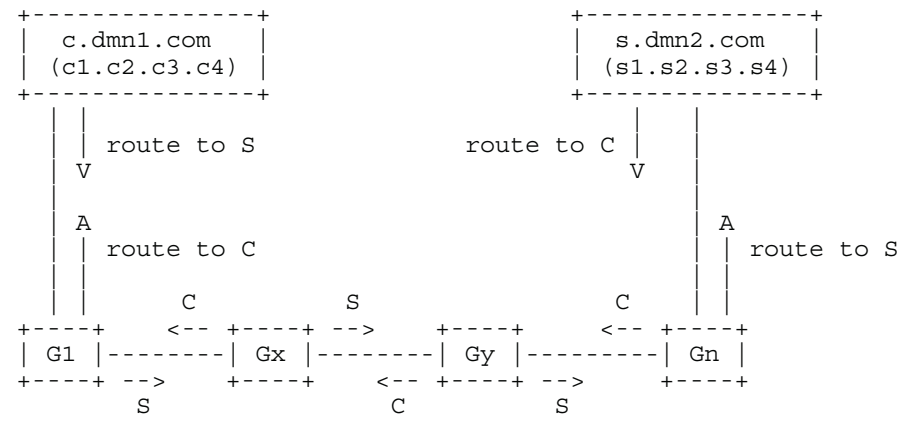
https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 167
Keysight, EX-1045
Page 167

intermediate name server). Ultimately, the client system obtains a valid IP address for s.dmn2.com. For simplicity, we assume the server has only one IP address.




Once the client system knows the IP address of the server system, it attempts to establish a connection to the standard FTP "control" TCP port on the server (port 21). For this to work, the client system must have a valid route to the server's IP address, and the server system must have a valid route to the client's IP address. All intermediate devices that behave like IP gateways must have valid routes for both the client and the server. If these devices perform packet filtering, they must ALL allow the specific type of traffic required between C and S for this specific application.






https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 168
Keysight, EX-1045
Page 168

The actual application work for the FTP session between the client and server is done with a bidirectional flow of TCP packets between the client's and server's IP addresses.

The FTP protocol uses a slightly complex protocol and TCP connection model which is, luckily, not important to the present discussion. This allows slightly shortening this document...

### 2.2 Requirements of direct communication

Based on the preceding discussion, it is possible to say that the following is required for a direct session between a client and server to be successful:

a) If the client uses the NAME of the server to reference it, the client must either have a hardcoded name-to-address binding for the server, or it must be able to resolve the server name (typically using DNS). In the case of DNS, this implies that the client and server must be part of the same DNS architecture or tree.

b) The client and server must be part of the same internetwork: the client must have a valid IP route towards the server, the server must have a valid IP route towards the client, and all intermediate IP gateways must have valid routes towards the client and server ("IP gateway" is the RFC standard terminology; people often use the term "IP router" in computer rooms).

c) If there are devices on the path between the client and server that perform packet filtering, all these devices must permit the forwarding of packets between the IP address of the client and the IP address of the server, at least for packets that fit the protocol model of the FTP application (TCP ports used, etc.).

## 3. Classical application proxies

A classical application proxy is a special program that knows one (or more) specific application protocols. Most application protocols are not symetric; one end is considered to be a "client", one end is a "server".

A classical application proxy implements both the "client" and "server" parts of an application protocol. In practice, it only needs to implement enough of the client and server protocols to accomplish the following:



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 169
Keysight, EX-1045
Page 169

a) accept client sessions and appear to them as a server;

b) receive from a client the name or address of the final target server (this needs to be passed over the "client-proxy" session in a way that is application-specific);

c) setup a session to the final server and appear to be a client from the server's point of view;

d) relay requests, responses, and data between the client and server;

e) perform access controls according to the proxy's design criteria (the main goal of the proxy, after all).

The functional goal of the proxy is to relay application data between clients and servers that may not have direct IP connectivity. The security goal of the proxy is to do checks and types of access controls that typical client and server software do not support or implement.

The following information will make it clear that classical proxies can offer many hidden benefits to the security-conscious network designer, at the cost of deploying client software with proxy capabilities or of educating the users on proxy use.

Client software issues are now easier to handle, given the increasing number of popular client applications (for Web, FTP, etc.) that offer proxy support. Designers developing new protocols are also more likely to plan proxy capability from the outset, to ensure their protocols can cross the many existing large corporate firewalls that are based at least in part on classical proxy technology.

### 3.1 Classical proxy session example

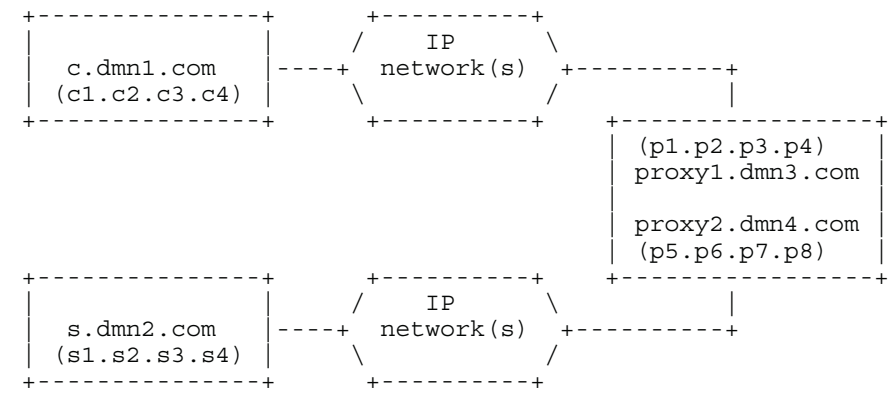
We will repeat our little analysis of an FTP session. This time, the FTP session is passing through a "classical" application proxy system. As is often the case (although not required), we will assume that the proxy system has two IP addresses, two network interfaces, and two DNS names.

The proxy system is running a special program which knows how to behave like an FTP client on one side, and like an FTP server on the other side. This program is what people call the "proxy". We will assume that the proxy program is listening to incoming requests on the standard FTP control port (21/tcp), although this is not always the case in practice.



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 170
Keysight, EX-1045
Page 170




The user starts an instance of an FTP client program on the client system "c.dmn1.com", and MUST specify that the target system is "proxy1.dmn3.com". On command-line systems, the user typically types:

```
ftp proxy1.dmn3.com
```

The client system needs to convert the proxy's name to an IP address (if the user directly specified the proxy by address, this step is not needed).

Converting the proxy name to an IP address requires work to be performed which ranges between two extremes:

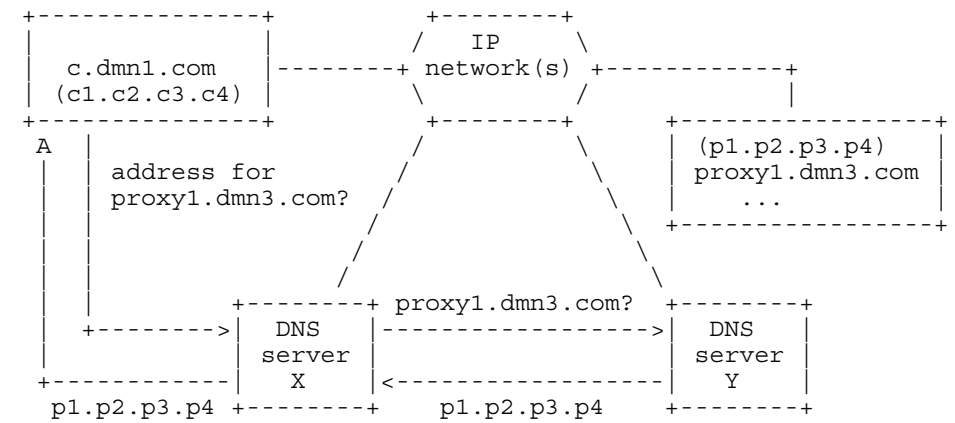
a) the client system has this name in its hosts file, or has local DNS caching capability and successfully retrieves the name of the proxy system in its cache. No network activity is performed to convert the name to an IP address.

b) the client system, in combination with DNS name servers, generate DNS queries that eventually propagate close to the root of the DNS tree and back down the proxy's DNS branch. Eventually, a DNS server which is authoritative for the proxy system's domain is queried and returns the IP address associated with "proxy1.dmn3.com" (depending on the case, it may return this to the client system directly or to an intermediate name server). Ultimately, the client system obtains a valid IP address for proxy1.dmn3.com.



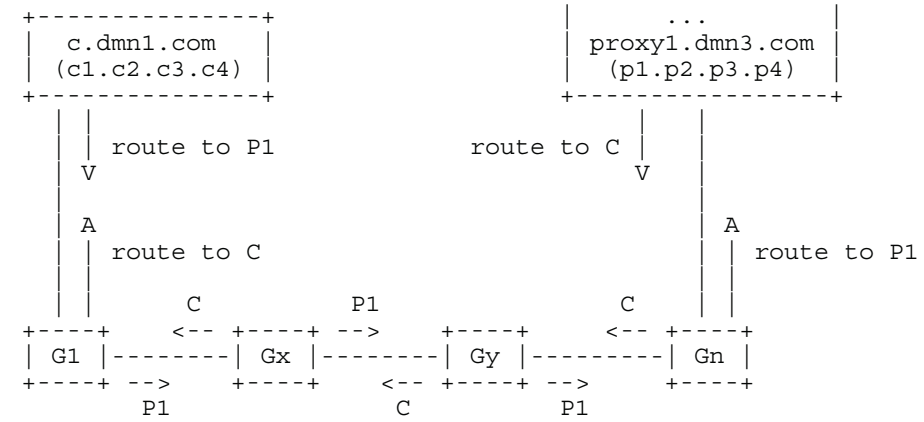
https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 171
Keysight, EX-1045
Page 171




Once the client system knows the IP address of the proxy system, it attempts to establish a connection to the standard FTP "control" TCP port on the proxy (port 21). For this to work, the client system must have a valid route to the proxy's IP address, and the proxy system must have a valid route to the client's IP address. All intermediate devices that behave like IP gateways must have valid routes to both the client and the proxy. If these devices perform packet filtering, they must ALL allow the specific type of traffic required between C and P1 for this specific application (FTP).

Finally, the proxy system must accept this incoming connection, based on the client's IP address (the purpose of the proxy is generally to do access control, after all).






https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 172
Keysight, EX-1045
Page 172

The actual application work for the FTP session between the client and proxy is done with a bidirectional flow of TCP packets between the client's and proxy's IP addresses.

For this to work, the proxy FTP application MUST fully support the FTP protocol and look identical to an FTP server from the client's point of view.

Once the client<->proxy session is established, the final target server name must be passed to the proxy, since, when using a "classical" application proxy, a way MUST be defined for the proxy to determine the final target system. This can be achieved in three ways:

a) The client system supplies the name or address of the final target system to the proxy in a method that is compatible with the specific application protocol being used (in our example, FTP). This is generally considered to be the main problem with classical proxies, since for each application being proxied, a method must be defined for passing the name or address of the final target system. This method must be compatible with every variant of client application that implements the protocol (i.e. the target-passing method must fit within the MINIMUM functionalities required by the specific application protocol).

   For the FTP protocol, the generally popular method for passing the final server name to the proxy is as follows:

   When the proxy prompts the FTP client for a username, the client specifies a string of the form:

```
target_username@target_system_name
or
target_username@target_ip_address
```

   The proxy will then know what is the final target system. The target_username (and the password supplied by the client) will be forwarded "as is" by the proxy to the final target system.

   A well-known example of an FTP proxy that behaves in this way is the "ftp-gw" program which is part of the Trusted Information System's firewall toolkit, available by anonymous FTP at ftp.tis.com. Several commercial firewalls also support this de-facto standard.



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 173
Keysight, EX-1045
Page 173

b) If there is only one possible final destination, the proxy may be configured to know this destination in advance. Since the IP address of the client system is known when the proxy must make this decision, the proxy can (if required) select a different destination based on the IP address of the client.

c) The client software may also support capabilities that allow it to present to the user the illusion of a direct session (the user just specifies the final target system, and the client software automatically handles the problem of reaching to the proxy system and passing the name or address of the final target system in whatever mutually-acceptable form).

   A well-known example of a system that provides modified client software, proxy software, and that provides the illusion of transparency is NEC's SOCKS system, available by anonymous FTP at ftp.nec.com.

   Alternatively, several FTP client applications support the "username@destination_host" de-facto standard implemented (for example) by the "ftp-gw" proxy application.

Once the FTP proxy application knows the name or IP address of the target system, it can choose to do two things:

a) Setup a session to the final target system, the more frequent case.

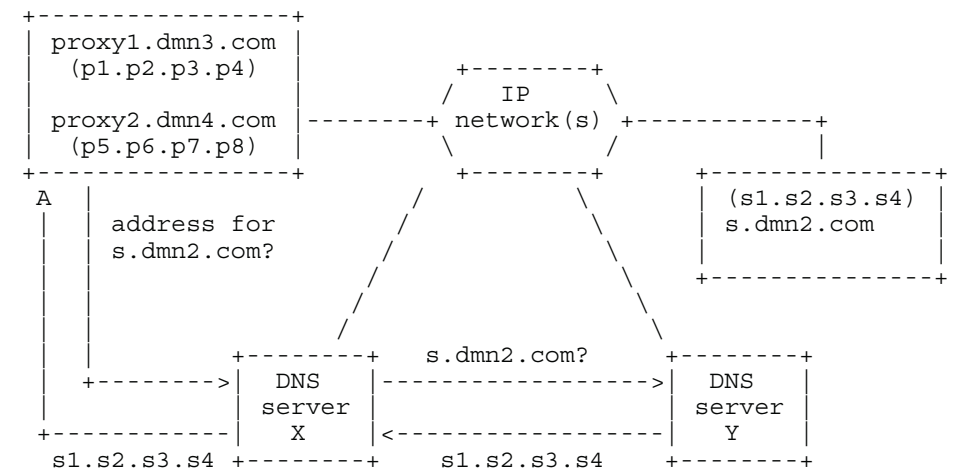
b) Decide (based on some internal configuration data) that it cannot reach the final target system directly, but must go through another proxy. This is rare today, but may become temporarily common due to the current shortage of IP network numbers which encourages organizations to deploy "hidden" network numbers which are already assigned elsewhere. Sessions between systems which have the same IP network number but which belong to different actual networks may require going through two proxy systems. This is discussed in more detail in section 3.2.6, "Interconnection of conflicting IP networks".

If the FTP proxy decides to connect directly to the target system, and what it has is the target system name, it will need to convert the target system name into an IP address. If this process involves DNS resolution, something like the following will happen:



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 174
Keysight, EX-1045
Page 174



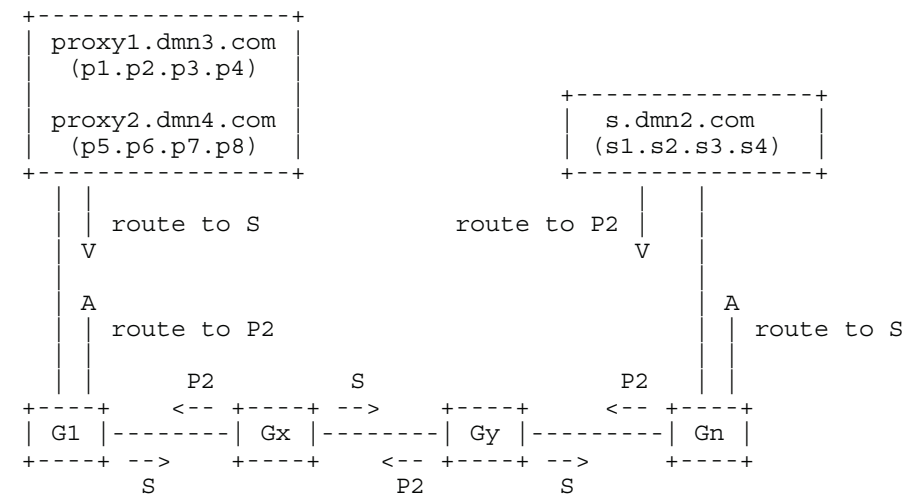

Once the proxy system knows the IP address of the server system, it attempts to establish a connection to the standard FTP "control" TCP port on the server (port 21). For this to work, the proxy system must have a valid route to the server's IP address, and the server system must have a valid route to at least one of the proxy's IP address. All intermediate devices that behave like IP gateways must have valid routes to both the proxy and the server. If these devices perform packet filtering, they must ALL allow the specific type of traffic required between the proxy and S for this specific application.



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 175
Keysight, EX-1045
Page 175

RFC 1919 Classical versus Transparent IP Proxies March 1996




The actual FTP application work between the proxy and server is done with a bidirectional flow of TCP packets between the proxy's and server's IP addresses.

What actually happens BETWEEN THE CLIENT AND SERVER? They both send replies and responses to the proxy, which forwards data to the "other" end. When one party opens a data connection and sends a PORT command to the proxy, the proxy allocates its own data connection and sends its PORT command to the "other" end. The proxy also copies data across the connections created in this way.

## 3.2 Characteristics of classical proxy configurations

Several IP internetworks may be linked using only classical proxy technology. It is currently popular to link two specific IP internetworks in this way: the Internet and some organization's "private" IP network. Such a proxy-based link is often the key component of a firewall.

When this is done, several benefits and problems are introduced for network administrators and users.

### 3.2.1 IP addressing and routing requirements.

The proxy system must be able to address all client and server systems to which it may provide service. It must also know valid IP routes to all these client and server systems.



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 176
Keysight, EX-1045
Page 176

Client and server systems must be able to address the proxy system, and must know a valid IP route to the proxy system. If the proxy system has several IP addresses (and often, several physical network interfaces), the client and server systems need only to be able to access ONE of the proxy system's IP addresses.

Note that client and server systems that use the proxy for communication DO NOT NEED valid IP addressing or routing information for systems that they reach through the proxy.

In this sense, it can be said that systems separated by a classical proxy are isolated from each other in an IP addressing sense and in an IP routing sense.

On the other hand, the classical proxy system (if running a standard TCP/IP software stack) needs to have a single coherent view of IP addressing and routing. If such a proxy system interconnects two IP networks and two systems use the same IP network/subnetwork number (one system on each network), the proxy will only be able to address one of the systems.

This restriction can be removed by chaining classical proxies (this is described later in section 3.2.6, "Interconnection of conflicting IP networks").

Using a classical proxy for interconnection of IP internetworks, it is also possible, with care, to achieve a desirable "fail-safe" feature: no valid routing entries need to exist for an internetwork which should be reached only through the proxy (routing updates that could add such entries shout be BLOCKED). If the proxy suddenly starts to behave like an IP router, only one-way attacks become possible.

In other words, assume an attacker has control of the remote internetwork and has found a way to cause the proxy to route IP packets, or has found a way to physically bypass the proxy.

The attacker may inject packets, but the attacked internal systems will be unable to reply to those packets. This certainly does not make attacks infeasible (as exemplified by certain holiday-period events in recent years), but it still makes attacks more difficult.



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 177
Keysight, EX-1045
Page 177

### 3.2.2 IP address hiding

Application "sessions" that go through a classical proxy are actually made of two complete sessions:

a) a session between the client and the proxy
b) a session between the proxy and the server

A device on the path sees only the client<->proxy traffic or the proxy<->server traffic, depending where it is located. If the two sessions actually pass through the same physical network, a device on that network may see both traffics, but may have difficulty establishing the relationship between the two sessions (depending on the specific application and activity level of the network).

A by-product of a classical proxy's behavior is commonly known as "address hiding". Equipments on some side of a classical proxy cannot easily determine what are the IP addresses used on another side of the proxy.

Address hiding is generally viewed as a Good Thing, since one of the purposes of deploying proxies is to disclose as little information about an internetwork as possible.

People who are in charge of gathering network statistics, and who do not have access to the proxy system's reports (if any) may consider address hiding to be a Bad Thing, since the proxy obscures the actual client/server relationships where the proxy was inserted. All IP activity originates and terminates on the proxy itself (or appears to do so).

In the same way, server software that accepts connections that have gone through a classical proxy do not see the IP address of the incoming client, unless this information is included in the application protocol (and even if it is, in many cases, the proxy will replace this information with its own address for the protocol to be consistent). This makes server access control unusable if it is based on client IP address checks.

### 3.2.3 DNS requirements

In most classical-proxy configurations, client systems pass the desired server name (or address) to the proxy system WITHOUT INTERPRETING IT. Because of this, the client system DOES NOT REQUIRE to be able to resolve the name of the server system in order to access it through a classical proxy. It only needs to be able to resolve the name of the proxy (if referencing the



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 178
Keysight, EX-1045
Page 178

proxy system by name).

Because of this, it can be said that a classical proxy system can offer DNS isolation. If two IP internetworks use completely separate DNS trees (each with their own DNS root servers), client software in one IP internetwork may still reference a server name in the other IP internetwork by passing its name to the classical proxy.

The classical proxy itself will not be able alone to resolve DNS names in both environments (if running standard DNS resolution software), since it will need to point to one or the other of the two DNS "universes".

A well-known technique called "split-brain DNS" can be used to relax this restriction somewhat, but such a technique ultimately involves prioritizing one DNS environment over another. If a DNS query can return a valid answer in both environments, only one of the answers will be found by the proxy.

### 3.2.4 Software requirements

A classical proxy application is a fairly simple piece of software, often simpler than either a real client implementation or a real server implementation. Such a program may run on any system that supports normal TCP/IP connections, and often does not require "system" or "superuser" privilege.

Classical proxy connections have no impact on normal server software; the proxy looks like a normal client in most respects except for its IP address and its "group" nature. All connections from the network on the other side of the proxy appear to come from the proxy, which poses problems if access control by client system is desired.

Normal client software may access a classical proxy if the user is willing or able to go through the extra steps necessary to indicate the final server to the proxy (whatever they are). Alternatively, modified (or newer) client software may be used that knows how to negotiate transparently with the proxy.

### 3.2.5 Impact of a classical proxy on packet filtering

If packet filtering is needed around a classical proxy, the packet filtering rules tend to be simplified, since the only traffic needed and allowed will originate from or terminate on the proxy (in an IP sense).



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 179
Keysight, EX-1045
Page 179

If the proxy starts behaving like an IP router, or if it is physically bypassed, such filtering rules, if deployed generally within an IP internetwork, will tend to prevent any direct traffic flow between the "internal" internetwork and "external" internetworks that are supposed to be only reachable through the application proxy.

### 3.2.6 Interconnection of conflicting IP networks

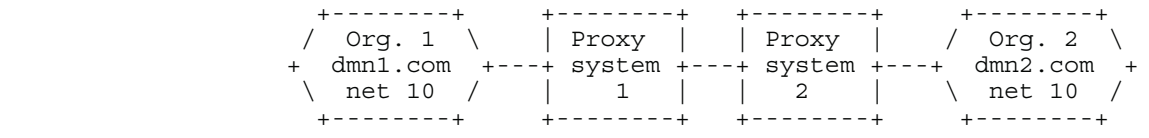
By chaining classical proxies, it is possible to achieve some interconnection of IP networks that have a high level of conflict. In practice, this type of setup resolves IP addressing conflicts much better than DNS conflicts. But DNS conflicts are currently less of a problem because the DNS "address space" is almost infinitely large (has anybody calculated the possible DNS address space based on the RFC-standard maximum host name length?).

Even though RFC 1597 was never more than an informational RFC, many organizations have been quietly following its suggestions, for lack of an easier solution. Now assume two organizations each use class A network number 10 on their network. Suddenly, they need to interconnect. What can they do?

First possibility: one side changes network number (not as hard as people think if properly planned, but this still represents some work)

Second possibility: they merge the two numbers by renumbering partially on each side to remove conflicts (actually harder to do, but has the political advantage that both sides have to do some work)

Third possibility: they communicate through chained classical proxies:




Both proxy 1 and 2 are standard systems running normal TCP/IP software stacks. Their configuration is not typical, however:



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 180
Keysight, EX-1045
Page 180

a) The link between proxy 1 and proxy 2 may use any IP network number that is not used (or not needed) on either side. Nothing on Org.1 and Org.2's networks need to have an IP route to this network.

b) Proxy 1 has an IP route for network 10 that points to Organization 1's network, and does DNS resolution (if required) using dmn1.com's name servers.

c) Proxy 2 has an IP route for network 10 that points to Organization 2's network, and does DNS resolution (if required) using dmn2.com's name servers.

d) Proxy 1 and proxy 2 only require a host IP route to each other for communication.

e) For this to be convenient, the classical proxy applications must support the automatic selection of a destination based on the client IP address.

f) On proxy system 1, the proxy software treats incoming sessions from proxy system 2 in the normal way: the "client" (proxy system 2) will be prompted in an application-specific way for the final destination. However, incoming sessions from Org.1 addresses are immediately and automatically forwarded to proxy system 2.

   Proxy system 2 is configured similarly (that is, connections coming from proxy 1 are prompted for a target server name, connections from Org.2 addresses are immediately and automatically forwarded to proxy 1.

From a user's point of view, the behavior of such a chained proxy system is not very different from a single classical application proxy:

a) A user on a client system with address 10.1.2.3 on Org.1's network wishes to do an anonymous FTP to "server.dmn2.com".

b) The user starts an FTP towards proxy 1. Proxy 1 sees an incoming connection from an address in network 10, so it immediately relays the connection to proxy 2.

c) Proxy 2 sees a connection coming from proxy 1, so it prompts the client. The user sees the username prompt



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 181
Keysight, EX-1045
Page 181

and types (assuming FTP proxies that behave like TIS's ftp-gw):

anonymous@server.dmn2.com

This will be resolved IN THE CONTEXT OF Org. 2'S NETWORK. The user can then complete the dialog and use the FTP connection.

d) Note that this setup will work even if the client and server have the EXACT SAME IP ADDRESS (10.1.2.3 in our example).

If the proxy applications support selecting another proxy based on the destination supplied by the client, and if DNS domains are unique, more than two conflicting IP networks can be linked in this way! Here is an example configuration:

a) Four IP networks that all use network 10 are linked by four proxy systems. The four proxy systems share a common, private IP network number and physical link (LAN or WAN).

b) A user on organization 1's network wishes to access a server on network 3. The user connects to its local proxy (proxy 1) and supplies that target system name.

c) Proxy 1 determines, based on a configuration rule, that the target system name is reachable by using proxy 3. So it connects to proxy 3 and passes the target system name.

d) Proxy 3 determines that the target system name is local (to itself) and connects to it directly.

Security Implications of chained proxies

Obviously, when such "chained" configurations are built, access control rules and logging based on a final-client/final-server combination are difficult to enforce, since the first proxy in the chain sees a final-client/proxy relationship and the last proxy in the chain sees a proxy/final-server relationship.

Doing better than this requires that the proxies be capable of passing the "original-client" and



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 182
Keysight, EX-1045
Page 182

"final-destination" information back and forth in the proxy chain for access control and/or logging purposes. This requires the proxies to trust each other, and requires the network path to be trusted (forging this information becomes an excellent attack).

Even if these problems were to be solved reliably, the original goal of the proxy chains was to solve an IP and possibly a DNS conflict. The "original-client" and "final-destination" values may not have the same meaning everywhere in the overall setup. Tagging the information with a "universe-name" may help, assuming it is possible to define unique universe names in the first place. Obviously this topic requires more study.

## 4. Transparent application proxies

The most visible problem of classical application proxies is the need for proxy-capable client programs and/or user education so that users know how to use the proxies.

When somebody thought of modifying proxies in such a way that normal user procedures and normal client applications would still be able to take advantage of the proxies, the transparent proxy was born.

A transparent application proxy is often described as a system that appears like a packet filter to clients, and like a classical proxy to servers. Apart from this important concept, transparent and classical proxies can do similar access control checks and can offer an equivalent level of security/robustness/performance, at least as far as the proxy itself is concerned.

The following information will make it clear that small organizations that wish to use proxy technology for protection, that wish to rely entirely on one proxy system for network perimeter security, that want a minimal (or zero) impact on user procedures, and that do not wish to bother with proxy-capable clients will tend to prefer transparent proxy technology.

Organizations with one or more of the following characteristics may prefer deploying classical proxy technology:

a) own a substantial internal IP router network, and wish to avoid adding "external" routes on the network
b) wish to deploy "defence in depth", such as internal firewalls, packet filtering on the internal network
c) wish to keep their DNS environment fully isolated from the "other side" of their proxy system, or that fear that their



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 183
Keysight, EX-1045
Page 183

RFC 1919 Classical versus Transparent IP Proxies March 1996

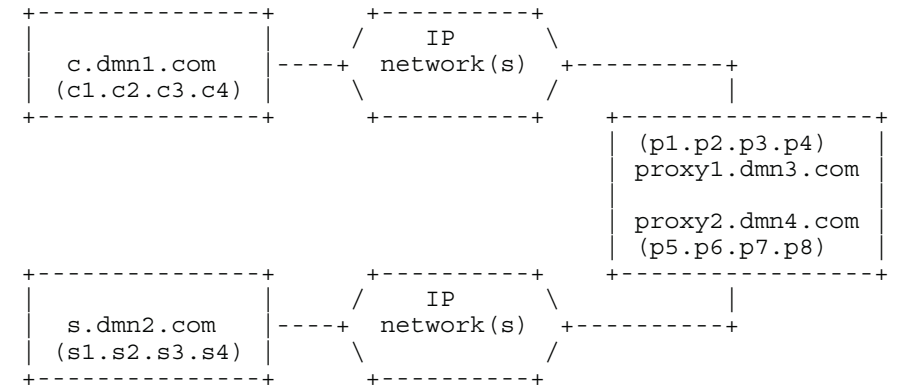
internal DNS servers may be vulnerable to data-driven attacks
d) use some IP networks that are in conflict with the "other side" of their proxy system
e) wish to use proxy applications that are easily portable to different operating system types and/or versions
f) wish to deploy multiple proxy systems interconnecting them to the SAME remote network without introducing dynamic routing for external routes on the internal network

### 4.1 Transparent proxy connection example

Let us go through an FTP sesssion again, through a "transparent" proxy this time. We assume that the proxy system has two IP addresses, two network interfaces, and two DNS names.

The proxy system is running a special program which knows how to behave like an FTP client on one side, and like an FTP server on the other side. This program is what people call the "proxy". This program, being a transparent proxy, also has a very special relationship with the TCP/IP implementation of the proxy system. This relationship may be built in several ways, we will describe only one such possible way.

We will assume that the proxy program is listening to incoming requests on the standard FTP control port (21/tcp), although this is not always the case in practice.






https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 184
Keysight, EX-1045
Page 184

The user starts an instance of an FTP client program on the client system "c.dmn1.com", and specifies a destination of "s.dmn2.com", just like if it was reachable directly. On command-line systems, the user typically types:

```
ftp s.dmn2.com
```

The client system needs to convert the server's name to an IP address (if the user directly specified the server by address, this step is not needed).

Converting the server name to an IP address requires work to be performed which ranges between two extremes:

a) the client system has this name in its hosts file, or has local DNS caching capability and successfully retrieves the name of the proxy system in its cache. No network activity is performed to convert the name to an IP address.

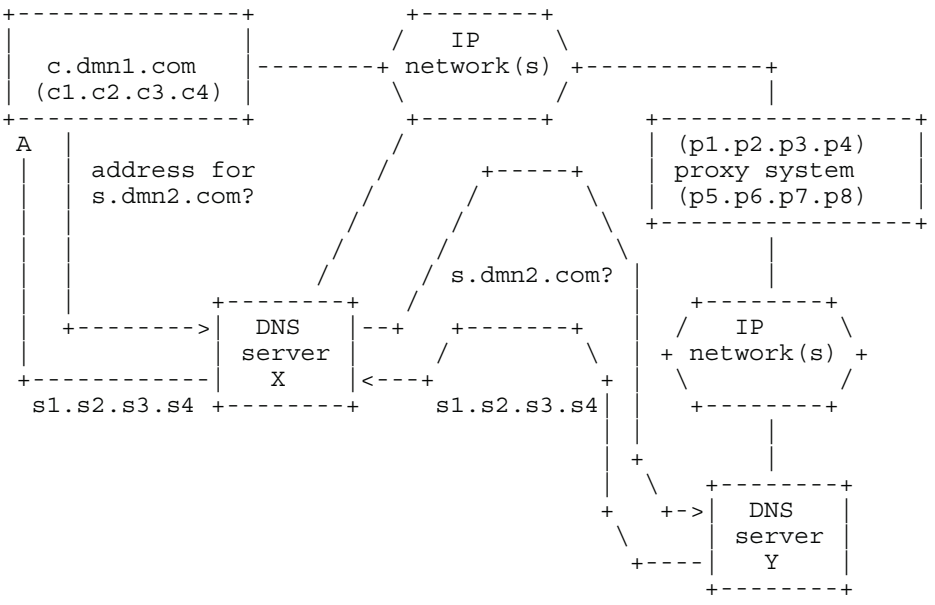
b) the client system, in combination with DNS name servers, generate DNS queries that eventually propagate close to the root of the DNS tree and back down the server's DNS branch. Eventually, a DNS server which is authoritative for the server system's domain is queried and returns the IP address associated with "s.dmn2.com" (depending on the case, it may return this to the client system directly or to an intermediate name server). Ultimately, the client system obtains a valid IP address for s.dmn2.com.



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 185
Keysight, EX-1045
Page 185

RFC 1919 Classical versus Transparent IP Proxies March 1996




NOTE: In practice, DNS servers that are authoritative for s.dmn2.com are highly likely to be located on the OTHER side of the proxy system. This means that DNS queries from the inside to the outside MUST be able to cross the proxy system. If the proxy system wishes to provide "address hiding", it must make these DNS queries (originating from the inside) appear to come from the proxy itself. This can be achieved by using a BIND-based DNS server (which has some proxy capabilities) or some simpler DNS proxy program. For full RFC compliance, the proxy system must be able to relay TCP-based queries just like UDP-based queries, since some client systems are rumored to ONLY use TCP for DNS queries.

The proxy system must be able to detect and block several classes of attacks based on DNS which (if nothing else) may cause denial of service:

a) attempts from the outside to return corrupt cache entries to an internal DNS server
b) attempts to return DNS bindings which have no relationship to the actual DNS query (some DNS servers are vulnerable to this). The attacker's goal may be to prime the cache of internal DNS servers with



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

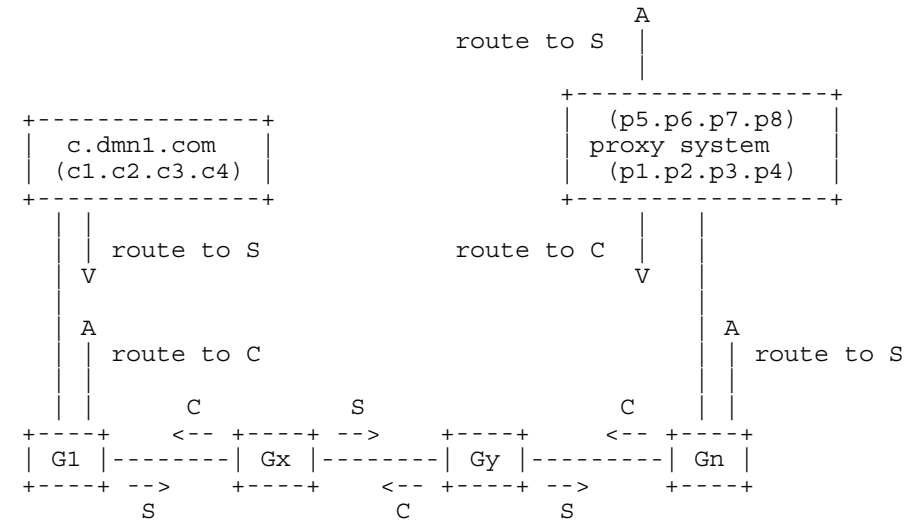
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 186
Keysight, EX-1045
Page 186

RFC 1919 Classical versus Transparent IP Proxies March 1996

interesting entries, including entries for internal DNS names that point to external IP addresses...

c) data-driven stuff similar in style to the "syslog buffer overrun" type attacks.

Once the client system knows the IP address of the server system, it attempts to establish a connection to the standard FTP "control" TCP port on the server (port 21). For this to work, the client system must have a valid route for the server's IP address THAT LEADS TO THE PROXY SYSTEM, and the proxy system must have a valid route for the client's IP address and the server's IP address. All intermediate devices that behave like IP gateways must have valid routes for the client, the server, and usually the proxy. If these devices perform packet filtering, they must ALL allow the specific type of traffic required between C and S for this specific application.




At the start of the FTP session, a TCP packet with a source address of C and a destination address of S travels to the proxy system, expecting to cross it just like a normal IP gateway.

This is when the transparent proxy shows its magic:

The proxy's TCP/IP software stack sees this incoming packets (and subsequent ones) for a destination address that is NOT one of its own addresses. Based on some criteria (a configuration file, for



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 187
Keysight, EX-1045
Page 187

example), it decides NOT to forward or drop the packet (which are the only two choices an RFC-standard TCP/IP implementation would have). The proxy system accepts the packet as if it was directed to one of its own IP addresses.

In our example, the incoming packet is a TCP packet. Since standard TCP/IP stacks store both a LOCAL and REMOTE IP address field for each TCP connection, the transparent proxy may set the LOCAL IP address field to the IP address that the client wants to reach (s1.s2.s3.s4 in our example). The standard TCP/IP stack probably needs to be modified to do this. UDP examples, although not connection-based, could be handled in similar ways.

Once this is done, the actual FTP proxy application is invoked since an incoming connection to TCP port 21 has occurred. It can determine what is the final target destination instantly, since the LOCAL IP address field of the connection contains the target server's IP address. There is no need for the proxy application to ask the client what is the final target system.

Since the FTP proxy application knows the IP address of the target system, it can choose to do two things:

a) Setup a session to the final target system, the more frequent case.

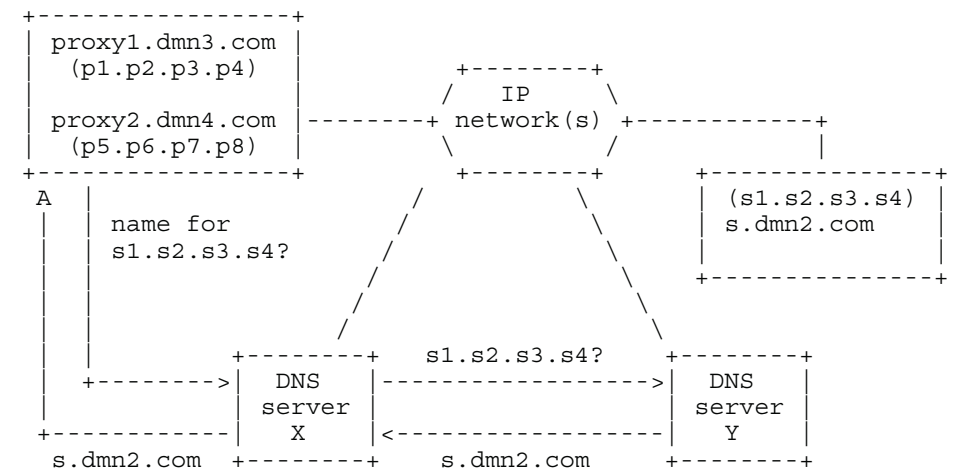
b) Decide (based on some internal configuration data) that it cannot reach the final target system directly, but must go through a "classical" proxy. This seems technically feasible, although no real transparent proxy system is known to offer this capability. The actual value of such a feature (if available) would need to be studied.

If the FTP proxy decides to connect directly to the target system, it has the target system's IP address. It may choose to do a reverse lookup on the target IP address to obtain a target system name (possibly needed for access control). If this process involves DNS resolution, something like the following will happen:



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 188
Keysight, EX-1045
Page 188

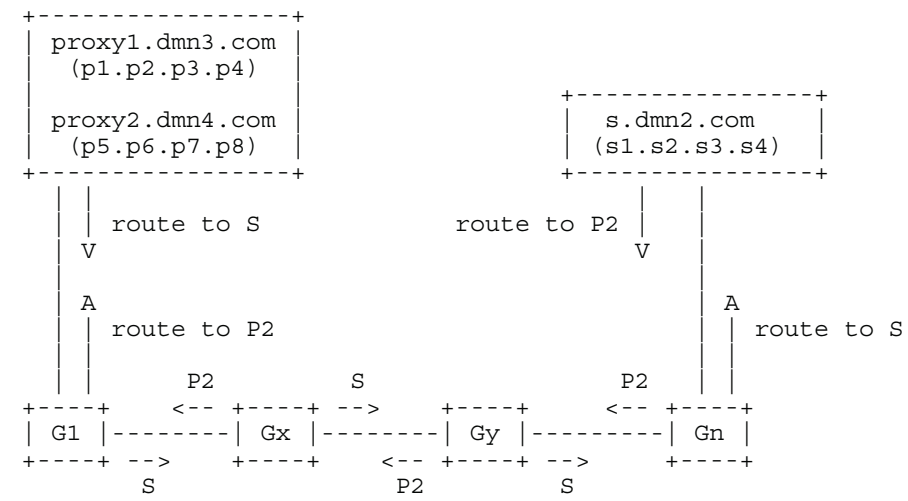



Once this is done and if the connection is allowed, the proxy attempts to establish a connection to the standard FTP "control" TCP port on the target server (port 21), using a technique identical to a "classical" proxy. For this to work, the proxy system must have a valid route to the server's IP address, and the server system must have a valid route to at least one of the proxy's IP address. All intermediate devices that behave like IP gateways must have valid routes to both the proxy and the server. If these devices perform packet filtering, they must ALL allow the specific type of traffic required between the proxy and S for this specific application.



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 189
Keysight, EX-1045
Page 189




The rest of the transparent proxy's operation is very similar to what would happen with a classical proxy.

## 4.2 Characteristics of transparent proxy configurations

Transparent proxy technology can be used to build the key component of a "firewall", in a way quite similar to the way classical proxy technology may be used. Several important details of the architecture must be different, however.

### 4.2.1 IP addressing and routing requirements

The transparent proxy system must be able to address all client and server systems to which it may provide service. It must also know valid IP routes to all these client and server systems.

Server systems must be able to address the proxy system, and must know a valid IP route to the proxy system. If the proxy system has several IP addresses (and often, several physical network interfaces), the server systems need only to be able to access ONE of the proxy system's IP addresses.

Client systems MUST HAVE valid IP addressing and routing information for systems that they reach through the proxy. For example, in the common case where a transparent proxy is being used to interconnect a private network and the Internet, the



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 190
Keysight, EX-1045
Page 190

private network will effectively need to use a default route that points to the transparent proxy system. This is a specific need of transparent proxy configurations.

Interconnecting two internetworks with multiple transparent proxies (for load sharing or fail-over) can be accomplished by using different techniques from what would be done for classical proxies:

a) with multiple classical proxies to the same remote network, clients can be configured to access different proxies manually, or DNS-based techniques, such as DNS load-balancing may be used to make clients access a different proxy at different times.

b) with multiple transparent proxies to the same remote network, the internal network must be able to provide dynamic routing towards the proxies (routing updates may need to be supplied by the proxies themselves). Client systems (depending on topology) may not need to see the route changes, but internal backbone routers probably do.

It is clear that internetworks linked by a transparent proxy cannot be fully isolated from each other in an IP addressing and routing sense. The network on which client systems are located must have effective valid routing entries to the remote internetwork; these routing entries must point to the proxy.

The transparent proxy system (if running a vaguely standard TCP/IP software stack) needs to have a single coherent view of IP addressing and routing. If a proxy system interconnects two IP networks and two systems use the same IP network/subnetwork number (one system on each internetwork), the proxy will only be able to address one of the systems. Even if the proxy is able to manage multiple conflicting IP universes (if, for example, one instance of a complete TCP/IP stack and its data structures is bound to each of the proxy network interfaces), the client systems will still have a problem: Why should it send packets with this network number to the proxy since this network number exists also on the internal internetwork?

Chaining transparent proxies does not seem at first glance to solve IP conflicts like it does for classical proxies.

From a "security" fail-safe point of view, the transparent proxy has an undesirable characteristic: the network being protected must have valid routing entries to the remote



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 191
Keysight, EX-1045
Page 191

network(s). If the proxy fails (starts behaving like a non-filtering IP router) or is physically bypassed, it is likely that the internal network will be immediately able to reply to "attacker" packets. The attacker does not need to modify routing tables or to spoof internal IP addresses.

This is important for organizations that do not wish to place ALL their confidence and protection into a proxy system (for whatever reason).

#### 4.2.2 IP address hiding

Application "sessions" that go through a transparent proxy are actually made of two complete sessions:

a) a session between the client and the address of the server, the session being "intercepted" by the proxy
b) a session between the proxy and the server

A device on the path sees either the client<->server traffic or the proxy<->server traffic, depending where it is located. The client<-"server" traffic is actually generated by the transparent proxy. The two sessions SHOULD NEVER pass through the same physical network, since in that case (due to the routing requirements) a total bypass of the proxy at the IP routing level may easily occur without being detectable.

Like classical proxies, transparent proxies accomplish a form of IP address hiding. Client IP addresses are hidden from the servers, since the servers see a session being initiated by the proxy. Server IP addresses are NOT hidden from the clients however, so that the illusion of transparency may be maintained.

This difference implies that internal (client-side) network statistics at the IP level will accurately reflect what outside destinations are being accessed. This can be useful for analyzing traffic patterns.

#### 4.2.3 DNS requirements

In transparent proxy configurations, client systems MUST be able to resolve server names belonging to remote networks. This is critical since the proxy will determine the target server from the destination IP address of the packets arriving from the client. Because of this, the "client" internetwork needs to have some form of DNS interconnection to the remote network. If internal client and name server IP addresses must be hidden



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 192
Keysight, EX-1045
Page 192

from the outside, these DNS queries must also be proxied.

Of course, remote host name/address relationships may be stored locally on the client systems, but it is well known that such an approach does not scale...

Because of this, it can be said that a transparent proxy system cannot offer DNS isolation. If two IP internetworks use completely separate DNS trees (each with their own DNS root servers), client software in one IP internetwork will not have a way of finding name/address relationships in the "other" DNS tree, and this information must be obtained in order to pass the desired address to the transparent proxy.

The classical proxy itself (if running standard DNS resolution software) will not be able alone to resolve DNS names in both environments, since it will need to point to one or the other of the two DNS "universes". Running multiple instances of DNS resolution software can allow the proxy to do this, however.

Because of the requirement placed on some form of DNS communication through the proxy, it is critical for the proxy to be able to protect ITSELF, internal clients, and internal name servers from data-driven attacks at the DNS level.

#### 4.2.4 Software requirements

The big advantage of transparent proxies is that normal client software may access remote servers with no modifications and no changes to user procedures.

The transparent proxy application itself may not need to be more complicated than a classical proxy application.

However, the proxy TCP/IP software stack cannot be a fully-standard (well, today's standard at least) TCP/IP stack, and requires specific extensions:

a) the ability to specify ranges of IP addresses that do not belong to the proxy itself, but for which "intercept" processing will occur: if packets arrive at the proxy with a destination IP address in those ranges, the IP stack will not forward or drop the packets; it will pass them up to application layers.

b) This mechanism requires that applications may obtain both the IP address from which the packets come, and the address to which the packets were going. Typical



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 193
Keysight, EX-1045
Page 193

RFC 1919 Classical versus Transparent IP Proxies March 1996

IP stacks should already have the fields available to store the info; it is a matter of updating them properly for these "intercepted" packets.

c) In the case of "intercepted" TCP packets, the TCP stack must support establishing TCP connections where the "local" IP address is not one of the proxy's IP address.

Any TCP/IP software implementation should be modifiable to perform these tasks. If a standard API becomes widely available to drive these extensions, and if this API is generally implemented, transparent proxies may become "portable" applications.

Until this occurs, it must be assumed that implementors have chosen different ways of accomplishing these functions, so that today's transparent proxy applications cannot be fully portable. It also remains to be seen how much work is needed to propagate these "extensions" to IPV6 software stacks.

### 4.2.5 Impact of a transparent proxy on packet filtering

The nature of a transparent proxy's functionality makes it difficult to deploy good packet filtering on the "inside" (or client-side) of the proxy. The proxy will "masquerade" as all the external systems. Because of this, internal packet filters WILL TYPICALLY NEED TO ALLOW IP traffic between internal and external IP addresses.

Depending on the actual security policy of the network, it may be possible to do filtering based on protocol type and/or on TCP bits (to filter based on connection setup direction), but filtering that blocks external IP addresses CANNOT be deployed.

If the proxy starts behaving like an IP router, or if physically bypassed, the practical limitations imposed on internal packet filtering imply that a lot of direct traffic between the inside and outside network will be allowed to flow. Furthermore, as we have seen previously, the internal network will have valid routing entries for external network numbers that point to the proxy. If multiple proxies have been deployed, the internal network may even HAVE TO TRUST routing updates generated by the proxy.

In general, if an internal network wishes to communicate with an external network through a transparent proxy, it MUST BE FUNDAMENTALLY DESIGNED TO COMMUNICATE DIRECTLY with that



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 194
Keysight, EX-1045
Page 194

external network. This is true at the IP addressing level, at the IP routing level, and at the DNS level. A proxy security failure in this type of environment is likely to result in immediate, total, and undetected accessibility of the internal network by the external network.

#### 4.2.6 Interconnection of conflicting IP networks

Unlike classical proxies, transparent proxies do not readily seem useful in solving IP addressing conflicts.

If two internetworks use the same network number(s), systems and routers in each internetwork will have valid routes to these network numbers. If these routes are changed to point to a transparent proxy, traffic that is meant to stay within the same internetwork would start to flow towards the proxy. The proxy will not be able to distinguish reliably between traffic between systems of the same internetwork, and traffic which is meant to cross the proxy.

A possible solution to this problem is described in section 6 of this document, "Improving transparent proxies".

## 5. Comparison chart of classical and transparent proxies

For those who do not like longish discussions of technical details, here is a one-page summary of the strengths/weaknesses/differences of classical and transparent proxies:

| Issue | Classical Proxy | Transparent Proxy |
|---|---|---|
| IP addressing | systems/gateways on each network need to address the proxy | systems/gateways on the "client" network need to address the remote networks |
| IP routing | systems/gateways on each network need a valid routing entry for the proxy | systems/gateways on the "client" network also need routing entries for remote entries |
| IP address hiding | systems on each side of the proxy are hidden from each other | systems on the "client" side are hidden from the other sides |



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 195
Keysight, EX-1045
Page 195

| | | |
|---|---|---|
| DNS | full isolation possible | resolution of outside names by inside systems is required |
| Proxy software requirements | runs on standard TCP/IP stack; can be portable | requires special TCP/IP stack; not 100% portable |
| Client software requirements | requires proxy-capable software or user education | nothing more than for a direct connection |
| User requirements | must use proxy-capable software or know how to use the proxy | nothing more than for a direct connection |
| Packet filtering | can filter out "external" addresses | cannot filter out "external" addresses |
| IP address conflict resolution | can be done with chained proxies that support auto-connect | no obvious way to get this to work |

## 6. Improving transparent proxies

The main issues with transparent proxies seem to revolve around the need to force "client" systems to directly access external addresses. To some people, this characteristic makes a transparent proxy look too much like a complicated packet filter. Can this problem be solved?

The first possibility that comes to mind is to use the flexibility of the DNS protocol to build new tricks. If we restrict the "internal" clients so that they MUST ALWAYS use DNS to resolve external host names AND THAT THEY MUST NEVER store permanent copies of external host addresses, the following technique would become theoretically possible (this is a very painful restriction, by the way):

a) arrange for all internal queries for external DNS names to go to the transparent proxy system (this can be done in a number of ways).

b) arrange for a routing entry to exist for a class A network number that is not used on the internal network. This IMPLIES that the internal network may not be part of the Internet. This routing entry will point to the transparent proxy system. For



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 196
Keysight, EX-1045
Page 196

the purpose of our discussion, this special network number will be X.0.0.0.

c) when an internal system generates a query for an external address, the query (if no answer is cached on the internal network) will reach the proxy system. Assuming the query is to obtain the IP address corresponding to a domain name, the proxy will go through the following algorithm:

- try to find a valid binding for this external domain name in its local cache

- if not found, it will ITSELF launch an external DNS query for the domain name. When (and if) it receives a valid reply, it creates a local cache entry containing:

    Time To Live of the reply
    Expiry Time of the cache entry (based on the current time)
    External domain name
    External IP address
    Dynamically allocated IP address of the form X.x1.x2.x3.

  and returns to the client the dynamically allocated IP address in the range X.0.0.0, NOT THE REAL ONE.

- the client may (or may not) store the IP address returned in its cache, and will then attempt to connect to the dynamically allocated IP address. This traffic will arrive at the proxy because of the routing setup.

- The transparent proxy intercepts the traffic and can identify the actual desired target it should connect to based on the dynamically allocated IP address supplied by the client.

Such an approach, if workable, could improve many characteristics of transparent proxies and may even make transparent proxies capable of handling IP network number conflicts.

However, the algorithm above leaves many difficult questions unsolved. Here is a list (by no means exhaustive) of these questions:

a) What is the percentage of client DNS resolver and DNS server implementations that conform to the RFC specifications in their handling of the Time-To-Live field?

b) How should the proxy handle other types of DNS queries for external domain names (inverse queries, queries for other resource record types)?



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 197
Keysight, EX-1045
Page 197

c) A client program may perform a DNS query once for an external name and then use the response for a long time (a large file transfer, or a permanent management session, for example). Should the proxy update the Expiry Time of cache entries based on the passing IP traffic, and if so, using what algorithm?

d) What new types of attacks would such a system introduce or make possible?

e) What data structures and resources (memory, disk) would be needed for an efficient implementation if the proxy must sustain a high rate of DNS queries for external names, and where a large number of different external names are referenced? The class A network number is used basically to reference cache entries. Would a 24-bit address space be sufficient for practical use?

f) What happens with the cache (and the functionality) if the proxy crashes or reboots?

Such a system would probably exhibit two types of intermittent failures:

a) a client system is still using the result of an external name query (some X.x1.x2.x3 address dynamically allocated by the proxy), but this binding no longer exists in the proxy's cache. The client attempts a connection to this address, which fails.

b) a client's name cache contains a binding for X.x1.x2.x3, but the proxy has already reused this address for a different external host name. The client attempts a connection to this address, sees no obvious errors, but reaches a different system from the expected one.

If somebody has ever implemented such a scheme, information and live experience in deploying it would be useful to the IP networking community.

## 7. Security Considerations

Most of this document is concerned with security implications of classical and transparent proxy technology.

## 8. Acknowledgements

I could not have written this document without the support of Digital Equipment Corporation for whom I work as a consultant.



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 198
Keysight, EX-1045
Page 198

## 9. References

[1] Cheswick, W., Bellovin, S., "Firewalls and Internet Security: Repelling the Wily Hacker", Addison-Wesley, 1994.

[2] Chapman, B., Zwicky, E., "Building Internet Firewalls", O'Reilly and Associates, Inc., September 1995.

[3] Comer, D., "Internetworking with TCP/IP volume 1: Principles, Protocols, and Architecture", Prentice-Hall, 1991.

[4] Comer, D., Stevens, D., "Internetworking with TCP/IP volume 2: "Design, Implementation, and Internals", Prentice-Hall, 1991.

[5] Postel, J., and J. Reynolds, "File Transfer Protocol (FTP)", STD 9, RFC 959, USC/Information Sciences Institute, October 1985.

[6] Huitema, C., "An experiment in DNS Based IP Routing", RFC 1383, INRIA, December 1992.

[7] Rekhter Y., Moskowitz B., Karrenberg D., de Groot, G., "Address Allocation for Private Internets", RFC 1597, IBM Corp., Chrysler Corp, RIPE NCC, March 1994.

[8] The TIS firewall toolkit's documentation, available on Trusted Information System's anonymous FTP site, ftp.tis.com.

[9] Many discussions in the last 18 months on the firewalls-digest mailing list maintained by Great Circle Associates. The archives of the list are maintained at ftp.greatcircle.com.

Author's Address

Marc Chatel
9, avenue Jean Monnet
74940 ANNECY-LE-VIEUX
FRANCE

EMail: mchatel@pax.eunet.ch
or at Digital Equipment:
Marc.Chatel@aeo.mts.dec.com



https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 199
Keysight, EX-1045
Page 199

https://web.archive.org/web/20070630145311/https://tools.ietf.org/html/rfc1919

Sourcefire: Security
for the real world.

- SUPPORT HOME
- DOWNLOADS
- LICENSE
- VRT NOTICES
- RULES
- ACCOUNT
- SUPPORT TICKET
- CONTACT US

Login

Password

Register | Forgot password?

Log in

Current Release

**DC :** 4.10

**3D Sensor :** 4.10

**RNA Software :** 4.9.1.7

Latest Updates

**2011-04-27**

SEU Update 2011-04-27: SEU 448

**2011-04-28**

VDB Update 2011-04-28: 66

Certifications

**2009-09-14**

ICSA Labs Network IPS Certification for Sourcefire 3D Sensor Family, version 4.8.0.2. Contact Technical Support for details.

Visio Stencil

Sourcefire 3D System.zip

Product Lifecycle

**It is Sourcefire's policy to provide support for the current and one minor version back of all products.**



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 201
Keysight, EX-1045
Page 201

Click here for a list of currently supported products.

**End-of-Life (EOL) Notice for 4.8.x:**
With the release of version 4.10 of the 3D System, Sourcefire will no longer support Defense Center 4.8.x, 3D Sensor 4.8.x, and prior versions.
Click here for a copy of Sourcefire's Product Lifecycle Policy.

**Hardware EOL Announcements**
Sourcefire periodically transitions its oldest supported appliances into an End-of-Life (EOL) phase.

Click here to view the March 2010 Hardware EOL Announcement.

Click here to view the October 2009 Hardware EOL Announcement.
Click here to view the February 2009 Hardware EOL Announcement.

**Sourcefire Visualizer EOL Announcement**
Effective February 2, 2009, Sourcefire Visualizer was transitioned into an EOL phase.
Click here for more information.
©2010 Sourcefire, Inc.
Terms of Use | Privacy Policy

IEN 149 J. Postel
RFC 765 ISI
June 1980

# FILE TRANSFER PROTOCOL

## INTRODUCTION

The objectives of FTP are 1) to promote sharing of files (computer programs and/or data), 2) to encourage indirect or implicit (via programs) use of remote computers, 3) to shield a user from variations in file storage systems among Hosts, and 4) to transfer data reliably and efficiently. FTP, though usable directly by a user at a terminal, is designed mainly for use by programs.

The attempt in this specification is to satisfy the diverse needs of users of maxi-Hosts, mini-Hosts, and TIPs, with a simple, and easily implemented protocol design.

This paper assumes knowledge of the following protocols described in the ARPA Internet Protocol Handbook.

The Transmission Control Protocol

The TELNET Protocol

## DISCUSSION

In this section, the terminology and the FTP model are discussed. The terms defined in this section are only those that have special significance in FTP. Some of the terminology is very specific to the FTP model; some readers may wish to turn to the section on the FTP model while reviewing the terminology.

### TERMINOLOGY

#### ASCII

The ASCII character set as defined in the ARPA Internet Protocol Handbook. In FTP, ASCII characters are defined to be the lower half of an eight-bit code set (i.e., the most significant bit is zero).

#### access controls

Access controls define users' access privileges to the use of a system, and to the files in that system. Access controls are necessary to prevent unauthorized or accidental use of files. It is the prerogative of a server-FTP process to invoke access controls.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 203
Keysight, EX-1045
Page 203

byte size

There are two byte sizes of interest in FTP: the logical byte size of the file, and the transfer byte size used for the transmission of the data. The transfer byte size is always 8 bits. The transfer byte size is not necessarily the byte size in which data is to be stored in a system, nor the logical byte size for interpretation of the structure of the data.

data connection

A simplex connection over which data is transferred, in a specified mode and type. The data transferred may be a part of a file, an entire file or a number of files. The path may be between a server-DTP and a user-DTP, or between two server-DTPs.

data port

The passive data transfer process "listens" on the data port for a connection from the active transfer process in order to open the data connection.

EOF

The end-of-file condition that defines the end of a file being transferred.

EOR

The end-of-record condition that defines the end of a record being transferred.

error recovery

A procedure that allows a user to recover from certain errors such as failure of either Host system or transfer process. In FTP, error recovery may involve restarting a file transfer at a given checkpoint.

FTP commands

A set of commands that comprise the control information flowing from the user-FTP to the server-FTP process.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 204
Keysight, EX-1045
Page 204

file

An ordered set of computer data (including programs), of arbitrary length, uniquely identified by a pathname.

mode

The mode in which data is to be transferred via the data connection. The mode defines the data format during transfer including EOR and EOF. The transfer modes defined in FTP are described in the Section on Transmission Modes.

NVT

The Network Virtual Terminal as defined in the TELNET Protocol.

NVFS

The Network Virtual File System. A concept which defines a standard network file system with standard commands and pathname conventions. FTP only partially implements the NVFS concept at this time.

page

A file may be structured as a set of independent parts called pages. FTP supports the transmission of discontinuous files as independent indexed pages.

pathname

Pathname is defined to be the character string which must be input to a file system by a user in order to identify a file. Pathname normally contains device and/or directory names, and file name specification. FTP does not yet specify a standard pathname convention. Each user must follow the file naming conventions of the file systems involved in the transfer.

record

A sequential file may be structured as a number of contiguous parts called records. Record structures are supported by FTP but a file need not have record structure.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 205
Keysight, EX-1045
Page 205

reply

A reply is an acknowledgment (positive or negative) sent from server to user via the TELNET connections in response to FTP commands. The general form of a reply is a completion code (including error codes) followed by a text string. The codes are for use by programs and the text is usually intended for human users.

server-DTP

The data transfer process, in its normal "active" state, establishes the data connection with the "listening" data port, sets up parameters for transfer and storage, and transfers data on command from its PI. The DTP can be placed in a "passive" state to listen for, rather than initiate a, connection on the data port.

server-FTP process

A process or set of processes which perform the function of file transfer in cooperation with a user-FTP process and, possibly, another server. The functions consist of a protocol interpreter (PI) and a data transfer process (DTP).

server-PI

The protocol interpreter "listens" on Port L for a connection from a user-PI and establishes a TELNET communication connection. It receives standard FTP commands from the user-PI, sends replies, and governs the server-DTP.

TELNET connections

The full-duplex communication path between a user-PI and a server-PI, operating according to the TELNET Protocol.

type

The data representation type used for data transfer and storage. Type implies certain transformations between the time of data storage and data transfer. The representation types defined in FTP are described in the Section on Establishing Data Connections.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 206
Keysight, EX-1045
Page 206

user

A human being or a process on behalf of a human being wishing to obtain file transfer service. The human user may interact directly with a server-FTP process, but use of a user-FTP process is preferred since the protocol design is weighted towards automata.

user-DTP

The data transfer process "listens" on the data port for a connection from a server-FTP process. If two servers are transferring data between them, the user-DTP is inactive.

user-FTP process

A set of functions including a protocol interpreter, a data transfer process and a user interface which together perform the function of file transfer in cooperation with one or more server-FTP processes. The user interface allows a local language to be used in the command-reply dialogue with the user.

user-PI

The protocol interpreter initiates the TELNET connection from its port U to the server-FTP process, initiates FTP commands, and governs the user-DTP if that process is part of the file transfer.

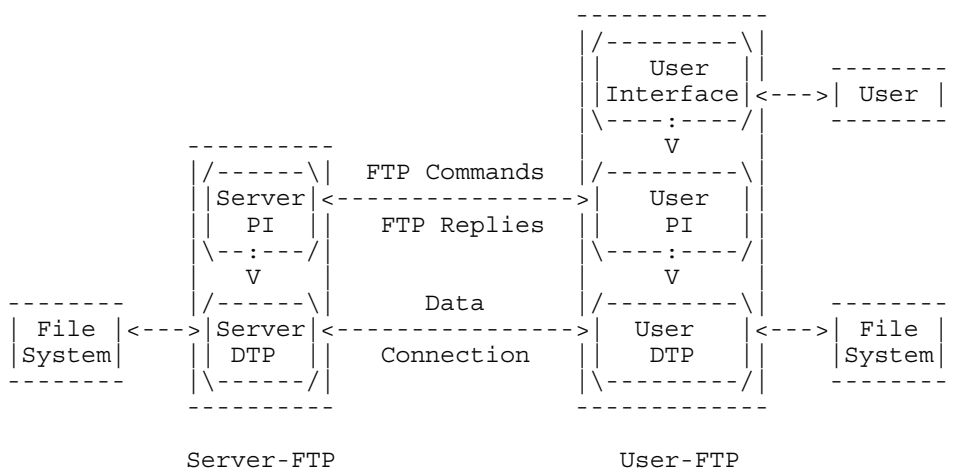


https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 207
Keysight, EX-1045
Page 207

THE FTP MODEL

With the above definitions in mind, the following model (shown in Figure 1) may be diagrammed for an FTP service.




NOTES: 1. The data connection may be used in either direction.
2. The data connection need not exist all of the time.

Figure 1 Model for FTP Use

In the model described in Figure 1, the user-protocol interpreter initiates the TELNET connection. At the initiation of the user, standard FTP commands are generated by the user-PI and transmitted to the server process via the TELNET connection. (The user may establish a direct TELNET connection to the server-FTP, from a TIP terminal for example, and generate standard FTP commands himself, bypassing the user-FTP process.) Standard replies are sent from the server-PI to the user-PI over the TELNET connection in response to the commands.

The FTP commands specify the parameters for the data connection (data port, transfer mode, representation type, and structure) and the nature of file system operation (store, retrieve, append, delete, etc.). The user-DTP or its designate should "listen" on the specified data port, and the server initiate the data connection and data transfer in accordance with the specified parameters. It should be noted that the data port need not be in



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 208
Keysight, EX-1045
Page 208

the same Host that initiates the FTP commands via the TELNET connection, but the user or his user-FTP process must ensure a "listen" on the specified data port. It should also be noted that the data connection may be used for simultaneous sending and receiving.

In another situation a user might wish to transfer files between two Hosts, neither of which is his local Host. He sets up TELNET connections to the two servers and then arranges for a data connection between them. In this manner control information is passed to the user-PI but data is transferred between the server data transfer processes. Following is a model of this server-server interaction.

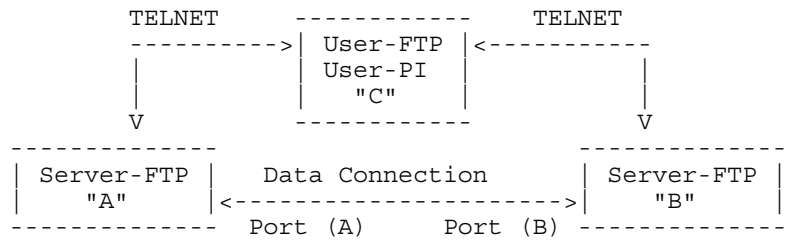



Figure 2

The protocol requires that the TELNET connections be open while data transfer is in progress. It is the responsibility of the user to request the closing of the TELNET connections when finished using the FTP service, while it is the server who takes the action. The server may abort data transfer if the TELNET connections are closed without command.

DATA TRANSFER FUNCTIONS

Files are transferred only via the data connection. The TELNET connection is used for the transfer of commands, which describe the functions to be performed, and the replies to these commands (see the Section on FTP Replies). Several commands are concerned with the transfer of data between Hosts. These data transfer commands include the MODE command which specify how the bits of the data are to be transmitted, and the STRUcture and TYPE commands, which are used to define the way in which the data are to be represented. The transmission and representation are basically independent but



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 209
Keysight, EX-1045
Page 209

"Stream" transmission mode is dependent on the file structure attribute and if "Compressed" transmission mode is used the nature of the filler byte depends on the representation type.

DATA REPRESENTATION AND STORAGE

Data is transferred from a storage device in the sending Host to a storage device in the receiving Host. Often it is necessary to perform certain transformations on the data because data storage representations in the two systems are different. For example, NVT-ASCII has different data storage representations in different systems. PDP-10's generally store NVT-ASCII as five 7-bit ASCII characters, left-justified in a 36-bit word. 360's store NVT-ASCII as 8-bit EBCDIC codes. Multics stores NVT-ASCII as four 9-bit characters in a 36-bit word. It may be desirable to convert characters into the standard NVT-ASCII representation when transmitting text between dissimilar systems. The sending and receiving sites would have to perform the necessary transformations between the standard representation and their internal representations.

A different problem in representation arises when transmitting binary data (not character codes) between Host systems with different word lengths. It is not always clear how the sender should send data, and the receiver store it. For example, when transmitting 32-bit bytes from a 32-bit word-length system to a 36-bit word-length system, it may be desirable (for reasons of efficiency and usefulness) to store the 32-bit bytes right-justified in a 36-bit word in the latter system. In any case, the user should have the option of specifying data representation and transformation functions. It should be noted that FTP provides for very limited data type representations. Transformations desired beyond this limited capability should be performed by the user directly.

Data representations are handled in FTP by a user specifying a representation type. This type may implicitly (as in ASCII or EBCDIC) or explicitly (as in Local byte) define a byte size for interpretation which is referred to as the "logical byte size." This has nothing to do with the byte size used for transmission over the data connection, called the "transfer byte size", and the two should not be confused. For example, NVT-ASCII has a logical byte size of 8 bits. If the type is Local byte, then the TYPE command has an obligatory second parameter specifying the logical byte size. The transfer byte size is always 8 bits.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 210
Keysight, EX-1045
Page 210

The types ASCII and EBCDIC also take a second (optional) parameter; this is to indicate what kind of vertical format control, if any, is associated with a file. The following data representation types are defined in FTP:

ASCII Format

This is the default type and must be accepted by all FTP implementations. It is intended primarily for the transfer of text files, except when both Hosts would find the EBCDIC type more convenient.

The sender converts the data from his internal character representation to the standard 8-bit NVT-ASCII representation (see the TELNET specification). The receiver will convert the data from the standard form to his own internal form.

In accordance with the NVT standard, the <CRLF> sequence should be used, where necessary, to denote the end of a line of text. (See the discussion of file structure at the end of the Section on Data Representation and Storage).

Using the standard NVT-ASCII representation means that data must be interpreted as 8-bit bytes.

The Format parameter for ASCII and EBCDIC types is discussed below.

EBCDIC Format

This type is intended for efficient transfer between Hosts which use EBCDIC for their internal character representation.

For transmission the data are represented as 8-bit EBCDIC characters. The character code is the only difference between the functional specifications of EBCDIC and ASCII types.

End-of-line (as opposed to end-of-record--see the discussion of structure) will probably be rarely used with EBCDIC type for purposes of denoting structure, but where it is necessary the <NL> character should be used.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 211
Keysight, EX-1045
Page 211

A character file may be transferred to a Host for one of three purposes: for printing, for storage and later retrieval, or for processing. If a file is sent for printing, the receiving Host must know how the vertical format control is represented. In the second case, it must be possible to store a file at a Host and then retrieve it later in exactly the same form. Finally, it ought to be possible to move a file from one Host to another and process the file at the second Host without undue trouble. A single ASCII or EBCDIC format does not satisfy all these conditions and so these types have a second parameter specifying one of the following three formats:

Non-print

This is the default format to be used if the second (format) parameter is omitted. Non-print format must be accepted by all FTP implementations.

The file need contain no vertical format information. If it is passed to a printer process, this process may assume standard values for spacing and margins.

Normally, this format will be used with files destined for processing or just storage.

TELNET Format Controls

The file contains ASCII/EBCDIC vertical format controls (i.e., <CR>, <LF>, <NL>, <VT>, <FF>) which the printer process will interpret appropriately. <CRLF>, in exactly this sequence, also denotes end-of-line.

Carriage Control (ASA)

The file contains ASA (FORTRAN) vertical format control characters. (See RFC 740 Appendix C and Communications of the ACM, Vol. 7, No. 10, 606 (Oct. 1964)). In a line or a record, formatted according to the ASA Standard, the first character is not to be printed. Instead it should be used to determine the vertical movement of the paper which should take place before the rest of the record is printed.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 212
Keysight, EX-1045
Page 212

The ASA Standard specifies the following control characters:

| Character | Vertical Spacing |
|---|---|
| blank | Move paper up one line |
| 0 | Move paper up two lines |
| 1 | Move paper to top of next page |
| + | No movement, i.e., overprint |

Clearly there must be some way for a printer process to distinguish the end of the structural entity. If a file has record structure (see below) this is no problem; records will be explicitly marked during transfer and storage. If the file has no record structure, the <CRLF> end-of-line sequence is used to separate printing lines, but these format effectors are overridden by the ASA controls.

Image

The data are sent as contiguous bits which, for transfer, are packed into the 8-bit transfer bytes. The receiving site must store the data as contiguous bits. The structure of the storage system might necessitate the padding of the file (or of each record, for a record-structured file) to some convenient boundary (byte, word or block). This padding, which must be all zeros, may occur only at the end of the file (or at the end of each record) and there must be a way of identifying the padding bits so that they may be stripped off if the file is retrieved. The padding transformation should be well publicized to enable a user to process a file at the storage site.

Image type is intended for the efficient storage and retrieval of files and for the transfer of binary data. It is recommended that this type be accepted by all FTP implementations.

Local byte Byte size

The data is transferred in logical bytes of the size specified by the obligatory second parameter, Byte size. The value of Byte size must be a decimal integer; there is no default value. The logical byte size is not necessarily the same as the transfer byte size. If there is a



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 213
Keysight, EX-1045
Page 213

difference in byte sizes, then the logical bytes should be packed contiguously, disregarding transfer byte boundaries and with any necessary padding at the end.

When the data reaches the receiving Host it will be transformed in a manner dependent on the logical byte size and the particular Host. This transformation must be invertible (that is an identical file can be retrieved if the same parameters are used) and should be well publicized by the FTP implementors.

For example, a user sending 36-bit floating-point numbers to a Host with a 32-bit word could send his data as Local byte with a logical byte size of 36. The receiving Host would then be expected to store the logical bytes so that they could be easily manipulated; in this example putting the 36-bit logical bytes into 64-bit double words should suffice.

Another example, a pair of hosts with a 36-bit word size may send data to one another in words by using TYPE L 36. The data would be sent in the 8-bit transmission bytes packed so that 9 transmission bytes carried two host words.

A note of caution about parameters: a file must be stored and retrieved with the same parameters if the retrieved version is to be identical to the version originally transmitted. Conversely, FTP implementations must return a file identical to the original if the parameters used to store and retrieve a file are the same.

In addition to different representation types, FTP allows the structure of a file to be specified. Three file structures are defined in FTP:

file-structure, where there is no internal structure and the file is considered to be a continuous sequence of data bytes,

record-structure, where the file is made up of sequential records,

and page-structure, where the file is made up of independent indexed pages.

File-structure is the default, to be assumed if the STRUcture command has not been used but both file and record structures must



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 214
Keysight, EX-1045
Page 214

be accepted for "text" files (i.e., files with TYPE ASCII or EBCDIC) by all FTP implementations. The structure of a file will affect both the transfer mode of a file (see the Section on Transmission Modes) and the interpretation and storage of the file.

The "natural" structure of a file will depend on which Host stores the file. A source-code file will usually be stored on an IBM 360 in fixed length records but on a PDP-10 as a stream of characters partitioned into lines, for example by <CRLF>. If the transfer of files between such disparate sites is to be useful, there must be some way for one site to recognize the other's assumptions about the file.

With some sites being naturally file-oriented and others naturally record-oriented there may be problems if a file with one structure is sent to a Host oriented to the other. If a text file is sent with record-structure to a Host which is file oriented, then that Host should apply an internal transformation to the file based on the record structure. Obviously this transformation should be useful but it must also be invertible so that an identical file may be retrieved using record structure.

In the case of a file being sent with file-structure to a record-oriented Host, there exists the question of what criteria the Host should use to divide the file into records which can be processed locally. If this division is necessary the FTP implementation should use the end-of-line sequence, <CRLF> for ASCII, or <NL> for EBCDIC text files, as the delimiter. If an FTP implementation adopts this technique, it must be prepared to reverse the transformation if the file is retrieved with file-structure.

Page Structure

To transmit files that are discontinuous FTP defines a page structure. Files of this type are sometimes know as "random access files" or even as "holey files". In these files there is sometimes other information associated with the file as a whole (e.g., a file descriptor), or with a section of the file (e.g., page access controls), or both. In FTP, the sections of the file are called pages.

To provide for various page sizes and associated information each page is sent with a page header. The page header has the following defined fields:



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 215
Keysight, EX-1045
Page 215

Header Length

The number of logical bytes in the page header including this byte. The minimum header length is 4.

Page Index

The logical page number of this section of the file. This is not the transmission sequence number of this page, but the index used to identify this page of the file.

Data Length

The number of logical bytes in the page data. The minimum data length is 0.

Page Type

The type of page this is. The following page types are defined:

0 = Last Page

This is used to indicate the end of a paged structured transmission. The header length must be 4, and the data length must be 0.

1 = Simple Page

This is the normal type for simple paged files with no page level associated control information. The header length must be 4.

2 = Descriptor Page

This type is used to transmit the descriptive information for the file as a whole.

3 = Access Controled Page

This is type includes an additional header field for paged files with page level access control information. The header length must be 5.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 216
Keysight, EX-1045
Page 216

Optional Fields

Further header fields may be used to supply per page control information, for example, per page access control.

All fields are one logical byte in length. The logical byte size is specified by the TYPE command.

ESTABLISHING DATA CONNECTIONS

The mechanics of transferring data consists of setting up the data connection to the appropriate ports and choosing the parameters for transfer. Both the user and the server-DTPs have a default data port. The user-process default data port is the same as the control connection port, i.e., U. The server-process default data port is the port adjacent to the control connection port, i.e., L-1.

The transfer byte size is 8-bit bytes. This byte size is relevant only for the actual transfer of the data; it has no bearing on representation of the data within a Host's file system.

The passive data transfer process (this may be a user-DTP or a second server-DTP) shall "listen" on the data port prior to sending a transfer request command. The FTP request command determines the direction of the data transfer. The server, upon receiving the transfer request, will initiate the data connection to the port. When the connection is established, the data transfer begins between DTP's, and the server-PI sends a confirming reply to the user-PI.

It is possible for the user to specify an alternate data port by use of the PORT command. He might want a file dumped on a TIP line printer or retrieved from a third party Host. In the latter case the user-PI sets up TELNET connections with both server-PI's. One server is then told (by an FTP command) to "listen" for a connection which the other will initiate. The user-PI sends one server-PI a PORT command indicating the data port of the other. Finally both are sent the appropriate transfer commands. The exact sequence of commands and replies sent between the user-controller and the servers is defined in the Section on FTP Replies.

In general it is the server's responsibility to maintain the data connection--to initiate it and to close it. The exception to this



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 217
Keysight, EX-1045
Page 217

is when the user-DTP is sending the data in a transfer mode that requires the connection to be closed to indicate EOF. The server MUST close the data connection under the following conditions:

1. The server has completed sending data in a transfer mode that requires a close to indicate EOF.

2. The server receives an ABORT command from the user.

3. The port specification is changed by a command from the user.

4. The TELNET connection is closed legally or otherwise.

5. An irrecoverable error condition occurs.

Otherwise the close is a server option, the exercise of which he must indicate to the user-process by an appropriate reply.

TRANSMISSION MODES

The next consideration in transferring data is choosing the appropriate transmission mode. There are three modes: one which formats the data and allows for restart procedures; one which also compresses the data for efficient transfer; and one which passes the data with little or no processing. In this last case the mode interacts with the structure attribute to determine the type of processing. In the compressed mode the representation type determines the filler byte.

All data transfers must be completed with an end-of-file (EOF) which may be explicitly stated or implied by the closing of the data connection. For files with record structure, all the end-of-record markers (EOR) are explicit, including the final one. For files transmitted in page structure a "last-page" page type is used.

NOTE: In the rest of this section, byte means "transfer byte" except where explicitly stated otherwise.

For the purpose of standardized transfer, the sending Host will translate his internal end of line or end of record denotation into the representation prescribed by the transfer mode and file structure, and the receiving Host will perform the inverse translation to his internal denotation. An IBM 360 record count field may not be recognized at another Host, so the end of record



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 218
Keysight, EX-1045
Page 218

information may be transferred as a two byte control code in Stream mode or as a flagged bit in a Block or Compressed mode descriptor. End of line in an ASCII or EBCDIC file with no record structure should be indicated by <CRLF> or <NL>, respectively. Since these transformations imply extra work for some systems, identical systems transferring non-record structured text files might wish to use a binary representation and stream mode for the transfer.

The following transmission modes are defined in FTP:

STREAM

The data is transmitted as a stream of bytes. There is no restriction on the representation type used; record structures are allowed.

In a record structured file EOR and EOF will each be indicated by a two-byte control code. The first byte of the control code will be all ones, the escape character. The second byte will have the low order bit on and zeros elsewhere for EOR and the second low order bit on for EOF; that is, the byte will have value 1 for EOR and value 2 for EOF. EOR and EOF may be indicated together on the last byte transmitted by turning both low order bits on, i.e., the value 3. If a byte of all ones was intended to be sent as data, it should be repeated in the second byte of the control code.

If the structure is file structure, the EOF is indicated by the sending Host closing the data connection and all bytes are data bytes.

BLOCK

The file is transmitted as a series of data blocks preceded by one or more header bytes. The header bytes contain a count field, and descriptor code. The count field indicates the total length of the data block in bytes, thus marking the beginning of the next data block (there are no filler bits). The descriptor code defines: last block in the file (EOF) last block in the record (EOR), restart marker (see the Section on Error Recovery and Restart) or suspect data (i.e., the data being transferred is suspected of errors and is not reliable). This last code is NOT intended for error control within FTP. It is motivated by the desire of sites



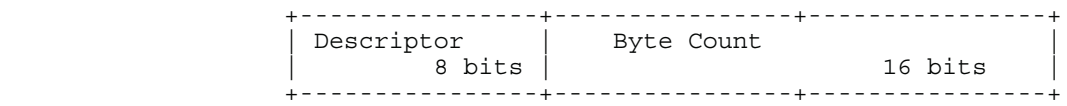
https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 219
Keysight, EX-1045
Page 219

exchanging certain types of data (e.g., seismic or weather data) to send and receive all the data despite local errors (such as "magnetic tape read errors"), but to indicate in the transmission that certain portions are suspect). Record structures are allowed in this mode, and any representation type may be used.

The header consists of the three bytes. Of the 24 bits of header information, the 16 low order bits shall represent byte count, and the 8 high order bits shall represent descriptor codes as shown below.

Block Header




The descriptor codes are indicated by bit flags in the descriptor byte. Four codes have been assigned, where each code number is the decimal value of the corresponding bit in the byte.

| Code | Meaning |
|---|---|
| 128 | End of data block is EOR |
| 64 | End of data block is EOF |
| 32 | Suspected errors in data block |
| 16 | Data block is a restart marker |

With this encoding more than one descriptor coded condition may exist for a particular block. As many bits as necessary may be flagged.

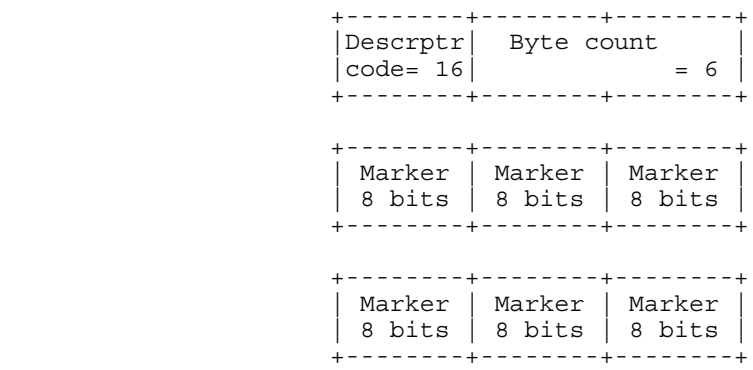
The restart marker is embedded in the data stream as an integral number of 8-bit bytes representing printable characters in the language being used over the TELNET connection (e.g., default--NVT-ASCII). <SP> (Space, in the appropriate language) must not be used WITHIN a restart marker.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 220
Keysight, EX-1045
Page 220

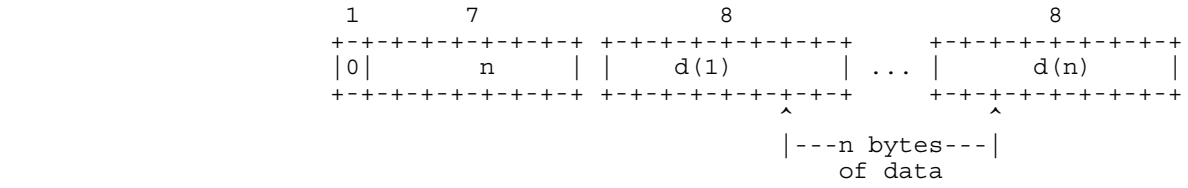
For example, to transmit a six-character marker, the following would be sent:




COMPRESSED

There are three kinds of information to be sent: regular data, sent in a byte string; compressed data, consisting of replications or filler; and control information, sent in a two-byte escape sequence. If n>0 bytes (up to 127) of regular data are sent, these n bytes are preceded by a byte with the left-most bit set to 0 and the right-most 7 bits containing the number n.

Byte string:




String of n data bytes d(1),..., d(n)
Count n must be positive.

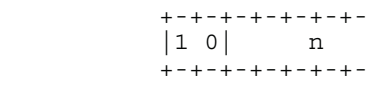
To compress a string of n replications of the data byte d, the following 2 bytes are sent:



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 221
Keysight, EX-1045
Page 221

Replicated Byte:




A string of n filler bytes can be compressed into a single byte, where the filler byte varies with the representation type. If the type is ASCII or EBCDIC the filler byte is <SP> (Space, ASCII code 32., EBCDIC code 64). If the type is Image or Local byte the filler is a zero byte.

Filler String:




The escape sequence is a double byte, the first of which is the escape byte (all zeros) and the second of which contains descriptor codes as defined in Block mode. The descriptor codes have the same meaning as in Block mode and apply to the succeeding string of bytes.

Compressed mode is useful for obtaining increased bandwidth on very large network transmissions at a little extra CPU cost. It can be most effectively used to reduce the size of printer files such as those generated by RJE Hosts.

## ERROR RECOVERY AND RESTART

There is no provision for detecting bits lost or scrambled in data transfer; this level of error control is handled by the TCP. However, a restart procedure is provided to protect users from gross system failures (including failures of a Host, an FTP-process, or the underlying network).

The restart procedure is defined only for the block and compressed modes of data transfer. It requires the sender of data to insert a special marker code in the data stream with some marker information. The marker information has meaning only to the sender, but must consist of printable characters in the default or negotiated language of the TELNET connection (ASCII or EBCDIC). The marker could represent a bit-count, a record-count, or any



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 222
Keysight, EX-1045
Page 222

other information by which a system may identify a data checkpoint. The receiver of data, if it implements the restart procedure, would then mark the corresponding position of this marker in the receiving system, and return this information to the user.

In the event of a system failure, the user can restart the data transfer by identifying the marker point with the FTP restart procedure. The following example illustrates the use of the restart procedure.

The sender of the data inserts an appropriate marker block in the data stream at a convenient point. The receiving Host marks the corresponding data point in its file system and conveys the last known sender and receiver marker information to the user, either directly or over the TELNET connection in a 110 reply (depending on who is the sender). In the event of a system failure, the user or controller process restarts the server at the last server marker by sending a restart command with server's marker code as its argument. The restart command is transmitted over the TELNET connection and is immediately followed by the command (such as RETR, STOR or LIST) which was being executed when the system failure occurred.

## FILE TRANSFER FUNCTIONS

The communication channel from the user-PI to the server-PI is established by a TCP connection from the user to a standard server port. The user protocol interpreter is responsible for sending FTP commands and interpreting the replies received; the server-PI interprets commands, sends replies and directs its DTP to set up the data connection and transfer the data. If the second party to the data transfer (the passive transfer process) is the user-DTP then it is governed through the internal protocol of the user-FTP Host; if it is a second server-DTP then it is governed by its PI on command from the user-PI. The FTP replies are discussed in the next section. In the description of a few of the commands in this section it is helpful to be explicit about the possible replies.

### FTP COMMANDS

#### ACCESS CONTROL COMMANDS

The following commands specify access control identifiers (command codes are shown in parentheses).



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 223
Keysight, EX-1045
Page 223

USER NAME (USER)

The argument field is a TELNET string identifying the user. The user identification is that which is required by the server for access to its file system. This command will normally be the first command transmitted by the user after the TELNET connections are made (some servers may require this). Additional identification information in the form of a password and/or an account command may also be required by some servers. Servers may allow a new USER command to be entered at any point in order to change the access control and/or accounting information. This has the effect of flushing any user, password, and account information already supplied and beginning the login sequence again. All transfer parameters are unchanged and any file transfer in progress is completed under the old account.

PASSWORD (PASS)

The argument field is a TELNET string identifying the user's password. This command must be immediately preceded by the user name command, and, for some sites, completes the user's identification for access control. Since password information is quite sensitive, it is desirable in general to "mask" it or suppress typeout. It appears that the server has no foolproof way to achieve this. It is therefore the responsibility of the user-FTP process to hide the sensitive password information.

ACCOUNT (ACCT)

The argument field is a TELNET string identifying the user's account. The command is not necessarily related to the USER command, as some sites may require an account for login and others only for specific access, such as storing files. In the latter case the command may arrive at any time.

There are reply codes to differentiate these cases for the automaton: when account information is required for login, the response to a successful PASSword command is reply code 332. On the other hand, if account information is NOT required for login, the reply to a successful PASSword command is 230; and if the account information is needed for a command issued later in the dialogue, the server should



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 224
Keysight, EX-1045
Page 224

return a 332 or 532 reply depending on whether he stores (pending receipt of the ACCounT command) or discards the command, respectively.

REINITIALIZE (REIN)

This command terminates a USER, flushing all I/O and account information, except to allow any transfer in progress to be completed. All parameters are reset to the default settings and the TELNET connection is left open. This is identical to the state in which a user finds himself immediately after the TELNET connection is opened. A USER command may be expected to follow.

LOGOUT (QUIT)

This command terminates a USER and if file transfer is not in progress, the server closes the TELNET connection. If file transfer is in progress, the connection will remain open for result response and the server will then close it. If the user-process is transferring files for several USERs but does not wish to close and then reopen connections for each, then the REIN command should be used instead of QUIT.

An unexpected close on the TELNET connection will cause the server to take the effective action of an abort (ABOR) and a logout (QUIT).

TRANSFER PARAMETER COMMANDS

All data transfer parameters have default values, and the commands specifying data transfer parameters are required only if the default parameter values are to be changed. The default value is the last specified value, or if no value has been specified, the standard default value as stated here. This implies that the server must "remember" the applicable default values. The commands may be in any order except that they must precede the FTP service request. The following commands specify data transfer parameters.

DATA PORT (PORT)

The argument is a HOST-PORT specification for the data port to be used in data connection. There defaults for both the user and server data ports, and under normal circumstances this command and its reply are not needed. If this command



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 225
Keysight, EX-1045
Page 225

is used the argument is the concatenation of a 32-bit internet host address and a 16-bit TCP port address. This address information is broken into 8-bit fields and the value of each field is transmitted as a decimal number (in character string representation). The fields are separated by commas. A port command would be:

PORT h1,h2,h3,h4,p1,p2

where, h1 is the high order 8 bits of the internet host address.

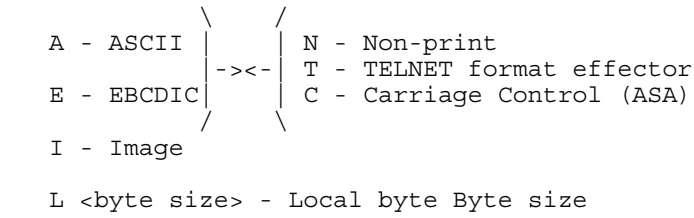
PASSIVE (PASV)

This command requests the server-DTP to "listen" on a data port (which is not its default data port) and to wait for a connection rather than initiate one upon receipt of a transfer command. The response to this command includes the host and port address this server is listening on.

REPRESENTATION TYPE (TYPE)

The argument specifies the representation type as described in the Section on Data Representation and Storage. Several types take a second parameter. The first parameter is denoted by a single TELNET character, as is the second Format parameter for ASCII and EBCDIC; the second parameter for local byte is a decimal integer to indicate Bytesize. The parameters are separated by a <SP> (Space, ASCII code 32.).

The following codes are assigned for type:




I - Image

L <byte size> - Local byte Byte size

The default representation type is ASCII Non-print. If the Format parameter is changed, and later just the first argument is changed, Format then returns to the Non-print default.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 226
Keysight, EX-1045
Page 226

FILE STRUCTURE (STRU)

The argument is a single TELNET character code specifying file structure described in the Section on Data Representation and Storage.

The following codes are assigned for structure:

F - File (no record structure)
R - Record structure
P - Page structure

The default structure is File.

TRANSFER MODE (MODE)

The argument is a single TELNET character code specifying the data transfer modes described in the Section on Transmission Modes.

The following codes are assigned for transfer modes:

S - Stream
B - Block
C - Compressed

The default transfer mode is Stream.

FTP SERVICE COMMANDS

The FTP service commands define the file transfer or the file system function requested by the user. The argument of an FTP service command will normally be a pathname. The syntax of pathnames must conform to server site conventions (with standard defaults applicable), and the language conventions of the TELNET connection. The suggested default handling is to use the last specified device, directory or file name, or the standard default defined for local users. The commands may be in any order except that a "rename from" command must be followed by a "rename to" command and the restart command must be followed by the interrupted service command. The data, when transferred in response to FTP service commands, shall always be sent over the data connection, except for certain informative replies. The following commands specify FTP service requests:



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 227
Keysight, EX-1045
Page 227

RETRIEVE (RETR)

This command causes the server-DTP to transfer a copy of the file, specified in the pathname, to the server- or user-DTP at the other end of the data connection. The status and contents of the file at the server site shall be unaffected.

STORE (STOR)

This command causes the server-DTP to accept the data transferred via the data connection and to store the data as a file at the server site. If the file specified in the pathname exists at the server site then its contents shall be replaced by the data being transferred. A new file is created at the server site if the file specified in the pathname does not already exist.

APPEND (with create) (APPE)

This command causes the server-DTP to accept the data transferred via the data connection and to store the data in a file at the server site. If the file specified in the pathname exists at the server site, then the data shall be appended to that file; otherwise the file specified in the pathname shall be created at the server site.

MAIL FILE (MLFL)

The intent of this command is to enable a user at the user site to mail data (in form of a file) to another user at the server site. It should be noted that the files to be mailed are transmitted via the data connection in ASCII or EBCDIC type. (It is the user's responsibility to ensure that the type is correct.) These files should be inserted into the destination user's mailbox by the server in accordance with serving Host mail conventions. The mail may be marked as sent from the particular user HOST and the user specified by the 'USER' command. The argument field may contain a Host system ident, or it may be empty. If the argument field is empty or blank (one or more spaces), then the mail is destined for a printer or other designated place for general delivery site mail.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 228
Keysight, EX-1045
Page 228

MAIL (MAIL)

This command allows a user to send mail that is NOT in a file over the TELNET connection. The argument field may contain system ident, or it may be empty. The ident is defined as above for the MLFL command. After the 'MAIL' command is received, the server is to treat the following lines as text of the mail sent by the user. The mail text is to be terminated by a line containing only a single period, that is, the character sequence "CRLF.CRLF". It is suggested that a modest volume of mail service should be free; i.e., it may be entered before a USER command.

MAIL SEND TO TERMINAL (MSND)

This command is like the MAIL command, except that the data is displayed on the addressed user's terminal, if such access is currently allowed, otherwise an error is returned.

MAIL SEND TO TERMINAL OR MAILBOX (MSOM)

This command is like the MAIL command, except that the data is displayed on the addressed user's terminal, if such access is currently allowed, otherwise the data is placed in the user's mailbox.

MAIL SEND TO TERMINAL AND MAILBOX (MSAM)

This command is like the MAIL command, except that the data is displayed on the addressed user's terminal, if such access is currently allowed, and, in any case, the data is placed in the user's mailbox.

MAIL RECIPIENT SCHEME QUESTION (MRSQ)

This FTP command is used to select a scheme for the transmission of mail to several users at the same host. The schemes are to list the recipients first, or to send the mail first.

MAIL RECIPIENT (MRCP)

This command is used to identify the individual recipients of the mail in the transmission of mail for multiple users at one host.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 229
Keysight, EX-1045
Page 229

ALLOCATE (ALLO)

This command may be required by some servers to reserve sufficient storage to accommodate the new file to be transferred. The argument shall be a decimal integer representing the number of bytes (using the logical byte size) of storage to be reserved for the file. For files sent with record or page structure a maximum record or page size (in logical bytes) might also be necessary; this is indicated by a decimal integer in a second argument field of the command. This second argument is optional, but when present should be separated from the first by the three TELNET characters <SP> R <SP>. This command shall be followed by a STORe or APPEnd command. The ALLO command should be treated as a NOOP (no operation) by those servers which do not require that the maximum size of the file be declared beforehand, and those servers interested in only the maximum record or page size should accept a dummy value in the first argument and ignore it.

RESTART (REST)

The argument field represents the server marker at which file transfer is to be restarted. This command does not cause file transfer but "spaces" over the file to the specified data checkpoint. This command shall be immediately followed by the appropriate FTP service command which shall cause file transfer to resume.

RENAME FROM (RNFR)

This command specifies the file which is to be renamed. This command must be immediately followed by a "rename to" command specifying the new file pathname.

RENAME TO (RNTO)

This command specifies the new pathname of the file specified in the immediately preceding "rename from" command. Together the two commands cause a file to be renamed.

ABORT (ABOR)

This command tells the server to abort the previous FTP service command and any associated transfer of data. The



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 230
Keysight, EX-1045
Page 230

abort command may require "special action", as discussed in the Section on FTP Commands, to force recognition by the server. No action is to be taken if the previous command has been completed (including data transfer). The TELNET connection is not to be closed by the server, but the data connection must be closed.

There are two cases for the server upon receipt of this command: (1) the FTP service command was already completed, or (2) the FTP service command is still in progress.

In the first case, the server closes the data connection (if it is open) and responds with a 226 reply, indicating that the abort command was successfully processed.

In the second case, the server aborts the FTP service in progress and closes the data connection, returning a 426 reply to indicate that the service request terminated in abnormally. The server then sends a 226 reply, indicating that the abort command was successfully processed.

DELETE (DELE)

This command causes the file specified in the pathname to be deleted at the server site. If an extra level of protection is desired (such as the query, "DO you really wish to delete?"), it should be provided by the user-FTP process.

CHANGE WORKING DIRECTORY (CWD)

This command allows the user to work with a different directory or dataset for file storage or retrieval without altering his login or accounting information. Transfer parameters are similarly unchanged. The argument is a pathname specifying a directory or other system dependent file group designator.

LIST (LIST)

This command causes a list to be sent from the server to the passive DTP. If the pathname specifies a directory, the server should transfer a list of files in the specified directory. If the pathname specifies a file then the server should send current information on the file. A null argument implies the user's current working or default



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 231
Keysight, EX-1045
Page 231

directory. The data transfer is over the data connection in type ASCII or type EBCDIC. (The user must ensure that the TYPE is appropriately ASCII or EBCDIC).

NAME-LIST (NLST)

This command causes a directory listing to be sent from server to user site. The pathname should specify a directory or other system-specific file group descriptor; a null argument implies the current directory. The server will return a stream of names of files and no other information. The data will be transferred in ASCII or EBCDIC type over the data connection as valid pathname strings separated by <CRLF> or <NL>. (Again the user must ensure that the TYPE is correct.)

SITE PARAMETERS (SITE)

This command is used by the server to provide services specific to his system that are essential to file transfer but not sufficiently universal to be included as commands in the protocol. The nature of these services and the specification of their syntax can be stated in a reply to the HELP SITE command.

STATUS (STAT)

This command shall cause a status response to be sent over the TELNET connection in the form of a reply. The command may be sent during a file transfer (along with the TELNET IP and Synch signals--see the Section on FTP Commands) in which case the server will respond with the status of the operation in progress, or it may be sent between file transfers. In the latter case the command may have an argument field. If the argument is a pathname, the command is analogous to the "list" command except that data shall be transferred over the TELNET connection. If a partial pathname is given, the server may respond with a list of file names or attributes associated with that specification. If no argument is given, the server should return general status information about the server FTP process. This should include current values of all transfer parameters and the status of connections.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 232
Keysight, EX-1045
Page 232

HELP (HELP)

This command shall cause the server to send helpful information regarding its implementation status over the TELNET connection to the user. The command may take an argument (e.g., any command name) and return more specific information as a response. The reply is type 211 or 214. It is suggested that HELP be allowed before entering a USER command. The server may use this reply to specify site-dependent parameters, e.g., in response to HELP SITE.

NOOP (NOOP)

This command does not affect any parameters or previously entered commands. It specifies no action other than that the server send an OK reply.

The File Transfer Protocol follows the specifications of the TELNET protocol for all communications over the TELNET connection. Since, the language used for TELNET communication may be a negotiated option, all references in the next two sections will be to the "TELNET language" and the corresponding "TELNET end of line code". Currently one may take these to mean NVT-ASCII and <CRLF>. No other specifications of the TELNET protocol will be cited.

FTP commands are "TELNET strings" terminated by the "TELNET end of line code". The command codes themselves are alphabetic characters terminated by the character <SP> (Space) if parameters follow and TELNET-EOL otherwise. The command codes and the semantics of commands are described in this section; the detailed syntax of commands is specified in the Section on Commands, the reply sequences are discussed in the Section on Sequencing of Commands and Replies, and scenarios illustrating the use of commands are provided in the Section on Typical FTP Scenarios.

FTP commands may be partitioned as those specifying access-control identifiers, data transfer parameters, or FTP service requests. Certain commands (such as ABOR, STAT, QUIT) may be sent over the TELNET connection while a data transfer is in progress. Some servers may not be able to monitor the TELNET and data connections simultaneously, in which case some special action will be necessary to get the server's attention. The exact form of the "special action" is undefined; but the following ordered format is tentatively recommended:



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 233
Keysight, EX-1045
Page 233

1. User system inserts the TELNET "Interrupt Process" (IP) signal in the TELNET stream.

2. User system sends the TELNET "Synch" signal

3. User system inserts the command (e.g., ABOR) in the TELNET stream.

4. Server PI,, after receiving "IP", scans the TELNET stream for EXACTLY ONE FTP command.

(For other servers this may not be necessary but the actions listed above should have no unusual effect.)

FTP REPLIES

Replies to File Transfer Protocol commands are devised to ensure the synchronization of requests and actions in the process of file transfer, and to guarantee that the user process always knows the state of the Server. Every command must generate at least one reply, although there may be more than one; in the latter case, the multiple replies must be easily distinguished. In addition, some commands occur in sequential groups, such as USER, PASS and ACCT, or RNFR and RNTO. The replies show the existence of an intermediate state if all preceding commands have been successful. A failure at any point in the sequence necessitates the repetition of the entire sequence from the beginning.

The details of the command-reply sequence are made explicit in a set of state diagrams below.

An FTP reply consists of a three digit number (transmitted as three alphanumeric characters) followed by some text. The number is intended for use by automata to determine what state to enter next; the text is intended for the human user. It is intended that the three digits contain enough encoded information that the user-process (the User-PI) will not need to examine the text and may either discard it or pass it on to the user, as appropriate. In particular, the text may be server-dependent, so there are likely to be varying texts for each reply code.

Formally, a reply is defined to contain the 3-digit code, followed by Space <SP>, followed by one line of text (where some maximum line length has been specified), and terminated by the TELNET end-of-line code. There will be cases, however, where the text is longer than a single line. In these cases the complete text must



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 234
Keysight, EX-1045
Page 234

be bracketed so the User-process knows when it may stop reading the reply (i.e. stop processing input on the TELNET connection) and go do other things. This requires a special format on the first line to indicate that more than one line is coming, and another on the last line to designate it as the last. At least one of these must contain the appropriate reply code to indicate the state of the transaction. To satisfy all factions it was decided that both the first and last line codes should be the same.

Thus the format for multi-line replies is that the first line will begin with the exact required reply code, followed immediately by a Hyphen, "-" (also known as Minus), followed by text. The last line will begin with the same code, followed immediately by Space <SP>, optionally some text, and the TELNET end-of-line code.

For example:

```
123-First line
Second line
  234 A line beginning with numbers
123 The last line
```

The user-process then simply needs to search for the second occurrence of the same reply code, followed by <SP> (Space), at the beginning of a line, and ignore all intermediary lines. If an intermediary line begins with a 3-digit number, the Server must pad the front to avoid confusion.

This scheme allows standard system routines to be used for reply information (such as for the STAT reply), with "artificial" first and last lines tacked on. In the rare cases where these routines are able to generate three digits and a Space at the beginning of any line, the beginning of each text line should be offset by some neutral text, like Space.

This scheme assumes that multi-line replies may not be nested. We have found that, in general, nesting of replies will not occur, except for random system messages (also called spontaneous replies) which may interrupt another reply. System messages (i.e. those not processed by the FTP server) will NOT carry reply codes and may occur anywhere in the command-reply sequence. They may be ignored by the User-process as they are only information for the human user.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 235
Keysight, EX-1045
Page 235

The three digits of the reply each have a special significance. This is intended to allow a range of very simple to very sophisticated response by the user-process. The first digit denotes whether the response is good, bad or incomplete. (Referring to the state diagram) an unsophisticated user-process will be able to determine its next action (proceed as planned, redo, retrench, etc.) by simply examining this first digit. A user-process that wants to know approximately what kind of error occurred (e.g. file system error, command syntax error) may examine the second digit, reserving the third digit for the finest gradation of information (e.g. RNTO command without a preceding RNFR.)

There are five values for the first digit of the reply code:

1yz Positive Preliminary reply

The requested action is being initiated; expect another reply before proceeding with a new command. (The user-process sending another command before the completion reply would be in violation of protocol; but server-FTP processes should queue any commands that arrive while a preceding command is in progress.) This type of reply can be used to indicate that the command was accepted and the user-process may now pay attention to the data connections, for implementations where simultaneous monitoring is difficult.

2yz Positive Completion reply

The requested action has been successfully completed. A new request may be initiated.

3yz Positive Intermediate reply

The command has been accepted, but the requested action is being held in abeyance, pending receipt of further information. The user should send another command specifying this information. This reply is used in command sequence groups.

4yz Transient Negative Completion reply

The command was not accepted and the requested action did not take place, but the error condition is temporary and the action may be requested again. The user should



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 236
Keysight, EX-1045
Page 236

return to the beginning of the command sequence, if any. It is difficult to assign a meaning to "transient", particularly when two distinct sites (Server and User-processes) have to agree on the interpretation. Each reply in the 4yz category might have a slightly different time value, but the intent is that the user-process is encouraged to try again. A rule of thumb in determining if a reply fits into the 4yz or the 5yz (Permanent Negative) category is that replies are 4yz if the commands can be repeated without any change in command form or in properties of the User or Server (e.g. the command is spelled the same with the same arguments used; the user does not change his file access or user name; the server does not put up a new implementation.)

5yz Permanent Negative Completion reply

The command was not accepted and the requested action did not take place. The User-process is discouraged from repeating the exact request (in the same sequence). Even some "permanent" error conditions can be corrected, so the human user may want to direct his User-process to reinitiate the command sequence by direct action at some point in the future (e.g. after the spelling has been changed, or the user has altered his directory status.)

The following function groupings are encoded in the second digit:

x0z Syntax - These replies refer to syntax errors, syntactically correct commands that don't fit any functional category, unimplemented or superfluous commands.

x1z Information - These are replies to requests for information, such as status or help.

x2z Connections - Replies referring to the TELNET and data connections.

x3z Authentication and accounting - Replies for the login process and accounting procedures.

x4z Unspecified as yet



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 237
Keysight, EX-1045
Page 237

x5z File system - These replies indicate the status of the Server file system vis-a-vis the requested transfer or other file system action.

The third digit gives a finer gradation of meaning in each of the function categories, specified by the second digit. The list of replies below will illustrate this. Note that the text associated with each reply is recommended, rather than mandatory, and may even change according to the command with which it is associated. The reply codes, on the other hand, must strictly follow the specifications in the last section; that is, Server implementations should not invent new codes for situations that are only slightly different from the ones described here, but rather should adapt codes already defined.

A command such as TYPE or ALLO whose successful execution does not offer the user-process any new information will cause a 200 reply to be returned. If the command is not implemented by a particular Server-FTP process because it has no relevance to that computer system, for example ALLO at a TOPS20 site, a Positive Completion reply is still desired so that the simple User-process knows it can proceed with its course of action. A 202 reply is used in this case with, for example, the reply text: "No storage allocation necessary." If, on the other hand, the command requests a non-site-specific action and is unimplemented, the response is 502. A refinement of that is the 504 reply for a command that IS implemented, but that requests an unimplemented parameter.

Reply Codes by Function Groups

200 Command okay
500 Syntax error, command unrecognized
[This may include errors such as command line too long.]
501 Syntax error in parameters or arguments
202 Command not implemented, superfluous at this site.
502 Command not implemented
503 Bad sequence of commands
504 Command not implemented for that parameter

110 Restart marker reply.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 238
Keysight, EX-1045
Page 238

```
   In this case the text is exact and not left to the
   particular implementation; it must read:
        MARK yyyy = mmmm
   where yyyy is User-process data stream marker, and mmmm
   server's equivalent marker.  (note the spaces between
   markers and "=".)
119 Terminal not available, will try mailbox.
211 System status, or system help reply
212 Directory status
213 File status
214 Help message
   (on how to use the server or the meaning of a particular
   non-standard command.  This reply is useful only to the
   human user.)
215 <scheme> is the preferred scheme.

120 Service ready in nnn minutes
220 Service ready for new user
221 Service closing TELNET connection
   (logged out if appropriate)
421 Service not available, closing TELNET connection.
   This may be a reply to any command if the service knows it
   must shut down.]
125 Data connection already open; transfer starting
225 Data connection open; no transfer in progress
425 Can't open data connection
226 Closing data connection;
   requested file action successful (for example, file transfer
   or file abort.)
426 Connection closed; transfer aborted.
227 Entering Passive Mode.  h1,h2,h3,h4,p1,p2

230 User logged in, proceed
530 Not logged in
331 User name okay, need password
332 Need account for login
532 Need account for storing files

150 File status okay; about to open data connection.
151 User not local; Will forward to <user>@<host>.
152 User Unknown; Mail will be forwarded by the operator.
250 Requested file action okay, completed.
350 Requested file action pending further information
450 Requested file action not taken:
   file unavailable (e.g. file busy)
550 Requested action not taken:
```



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 239
Keysight, EX-1045
Page 239

```
      file unavailable (e.g. file not found, no access)
   451 Requested action aborted: local error in processing
   551 Requested action aborted: page type unknown
   452 Requested action not taken:
      insufficient storage space in system
   552 Requested file action aborted:
      exceeded storage allocation (for current directory or
      dataset)
   553 Requested action not taken:
      file name not allowed
   354 Start mail input; end with <CR><LF>.<CR><LF>


Numeric Order List of Reply Codes

   110 Restart marker reply.
      In this case the text is exact and not left to the
      particular implementation; it must read:
           MARK yyyy = mmmm
      where yyyy is User-process data stream marker, and mmmm
      server's equivalent marker.  (note the spaces between
      markers and "=".)
   119 Terminal not available, will try mailbox.
   120 Service ready in nnn minutes
   125 Data connection already open; transfer starting
   150 File status okay; about to open data connection.
   151 User not local; Will forward to <user>@<host>.
   152 User Unknown; Mail will be forwarded by the operator.
   200 Command okay
   202 Command not implemented, superfluous at this site.
   211 System status, or system help reply
   212 Directory status
   213 File status
   214 Help message
      (on how to use the server or the meaning of a particular
      non-standard command.  This reply is useful only to the
      human user.)
   215 <scheme> is the preferred scheme.
   220 Service ready for new user
   221 Service closing TELNET connection
      (logged out if appropriate)
   225 Data connection open; no transfer in progress
   226 Closing data connection;
      requested file action successful (for example, file transfer
      or file abort.)
   227 Entering Passive Mode.  h1,h2,h3,h4,p1,p2
```



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 240
Keysight, EX-1045
Page 240

```
230 User logged in, proceed
250 Requested file action okay, completed.
331 User name okay, need password
332 Need account for login
350 Requested file action pending further information
354 Start mail input; end with <CR><LF>.<CR><LF>
421 Service not available, closing TELNET connection.
   This may be a reply to any command if the service knows it
   must shut down.]
425 Can't open data connection
426 Connection closed; transfer aborted.
450 Requested file action not taken:
   file unavailable (e.g. file busy)
451 Requested action aborted: local error in processing
452 Requested action not taken:
   insufficient storage space in system
500 Syntax error, command unrecognized
   [This may include errors such as command line too long.]
501 Syntax error in parameters or arguments
502 Command not implemented
503 Bad sequence of commands
504 Command not implemented for that parameter
530 Not logged in
532 Need account for storing files
550 Requested action not taken:
   file unavailable (e.g. file not found, no access)
551 Requested action aborted: page type unknown
552 Requested file action aborted:
   exceeded storage allocation (for current directory or
   dataset)
553 Requested action not taken:
   file name not allowed
```



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 241
Keysight, EX-1045
Page 241

DECLARATIVE SPECIFICATIONS

MINIMUM IMPLEMENTATION

In order to make FTP workable without needless error messages, the following minimum implementation is required for all servers:

```
TYPE - ASCII Non-print
MODE - Stream
STRUCTURE - File, Record
COMMANDS - USER, QUIT, PORT,
           TYPE, MODE, STRU,
             for the default values
           RETR, STOR,
           NOOP.
```

The default values for transfer parameters are:

```
TYPE - ASCII Non-print
MODE - Stream
STRU - File
```

All Hosts must accept the above as the standard defaults.

CONNECTIONS

The server protocol interpreter shall "listen" on Port L. The user or user protocol interpreter shall initiate the full-duplex TELNET connection. Server- and user- processes should follow the conventions of the TELNET protocol as specified in the ARPA Internet Protocol Handbook. Servers are under no obligation to provide for editing of command lines and may specify that it be done in the user Host. The TELNET connection shall be closed by the server at the user's request after all transfers and replies are completed.

The user-DTP must "listen" on the specified data port; this may be the default user port (U) or a port specified in the PORT command. The server shall initiate the data connection from his own default data port (L-1) using the specified user data port. The direction of the transfer and the port used will be determined by the FTP service command.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 242
Keysight, EX-1045
Page 242

When data is to be transferred between two servers, A and B (refer to Figure 2), the user-PI, C, sets up TELNET connections with both server-PI's. One of the servers, say A, is then sent a PASV command telling him to "listen" on his data port rather than initiate a connection when he receives a transfer service command. When the user-PI receives an acknowledgment to the PASV command, which includes the identity of the host and port being listened on, the user-PI then sends A's port, a, to B in a PORT command; a reply is returned. The user-PI may then send the corresponding service commands to A and B. Server B initiates the connection and the transfer proceeds. The command-reply sequence is listed below where the messages are vertically synchronous but horizontally asynchronous:

```
User-PI - Server A                User-PI - Server B
------------------                ------------------

C->A : Connect                    C->B : Connect
C->A : PASV
A->C : 227 Entering Passive Mode. A1,A2,A3,A4,a1,a2
                                  C->B : PORT A1,A2,A3,A4,a1,a2
                                  B->C : 200 Okay
C->A : STOR                       C->B : RETR
           B->A : Connect to HOST-A, PORT-a
```

The data connection shall be closed by the server under the conditions described in the Section on Establishing Data Connections. If the server wishes to close the connection after a transfer where it is not required, he should do so immediately after the file transfer is completed. He should not wait until after a new transfer command is received because the user-process will have already tested the data connection to see if it needs to do a "listen"; (recall that the user must "listen" on a closed data port BEFORE sending the transfer request). To prevent a race condition here, the server sends a reply (226) after closing the data connection (or if the connection is left open, a "file transfer completed" reply (250) and the user-PI should wait for one of these replies before issuing a new transfer command.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 243
Keysight, EX-1045
Page 243

COMMANDS

The commands are TELNET character string transmitted over the TELNET connections as described in the Section on FTP Commands. The command functions and semantics are described in the Section on Access Control Commands, Transfer Parameter Commands, FTP Service Commands, and Miscellaneous Commands. The command syntax is specified here.

The commands begin with a command code followed by an argument field. The command codes are four or fewer alphabetic characters. Upper and lower case alphabetic characters are to be treated identically. Thus any of the following may represent the retrieve command:

RETR Retr retr ReTr rETr

This also applies to any symbols representing parameter values, such as A or a for ASCII TYPE. The command codes and the argument fields are separated by one or more spaces.

The argument field consists of a variable length character string ending with the character sequence <CRLF> (Carriage Return, Linefeed) for NVT-ASCII representation; for other negotiated languages a different end of line character might be used. It should be noted that the server is to take NO action until the end of line code is received.

The syntax is specified below in NVT-ASCII. All characters in the argument field are ASCII characters including any ASCII represented decimal integers. Square brackets denote an optional argument field. If the option is not taken, the appropriate default is implied.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 244
Keysight, EX-1045
Page 244

The following are the FTP commands:

```
   USER <SP> <username> <CRLF>
   PASS <SP> <password> <CRLF>
   ACCT <SP> <account information> <CRLF>
   REIN <CRLF>
   QUIT <CRLF>
   PORT <SP> <Host-port> <CRLF>
   PASV <CRLF>
   TYPE <SP> <type code> <CRLF>
   STRU <SP> <structure code> <CRLF>
   MODE <SP> <mode code> <CRLF>
   RETR <SP> <pathname> <CRLF>
   STOR <SP> <pathname> <CRLF>
   APPE <SP> <pathname> <CRLF>
   MLFL [<SP> <ident>] <CRLF>
   MAIL [<SP> <ident>] <CRLF>
   MSND [<SP> <ident>] <CRLF>
   MSOM [<SP> <ident>] <CRLF>
   MSAM [<SP> <ident>] <CRLF>
   MRSQ [<SP> <scheme>] <CRLF>
   MRCP <SP> <ident> <CRLF>
   ALLO <SP> <decimal integer>
       [<SP> R <SP> <decimal integer>] <CRLF>
   REST <SP> <marker> <CRLF>
   RNFR <SP> <pathname> <CRLF>
   RNTO <SP> <pathname> <CRLF>
   ABOR <CRLF>
   DELE <SP> <pathname> <CRLF>
   CWD <SP> <pathname> <CRLF>
   LIST [<SP> <pathname>] <CRLF>
   NLST [<SP> <pathname>] <CRLF>
   SITE <SP> <string> <CRLF>
   STAT [<SP> <pathname>] <CRLF>
   HELP [<SP> <string>] <CRLF>
   NOOP <CRLF>
```



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 245
Keysight, EX-1045
Page 245

The syntax of the above argument fields (using BNF notation where applicable ) is:

```
<username> ::= <string>
<password> ::= <string>
<account information> ::= <string>
<string> ::= <char> | <char><string>
<char> ::= any of the 128 ASCII characters except <CR> and <LF>
<marker> ::= <pr string>
<pr string> ::= <pr char> | <pr char><pr string>
<pr char> ::= printable characters, any
              ASCII code 33 through 126
<byte size> ::= any decimal integer 1 through 255
<Host-port> ::= <Host-number>,<Port-number>
<Host-number> ::= <number>,<number>,<number>,<number>
<Port-number> ::= <number>,<number>
<number> ::= any decimal integer 0 through 255
<ident> ::= <string>
<scheme> ::= R | T | ?
<form code> ::= N | T | C
<type code> ::= A [<SP> <form code>]
              | E [<SP> <form code>]
              | I
              | L <SP> <byte size>
<structure code> ::= F | R | P
<mode code> ::= S | B | C
<pathname> ::= <string>
```



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 246
Keysight, EX-1045
Page 246

SEQUENCING OF COMMANDS AND REPLIES

The communication between the user and server is intended to be an alternating dialogue. As such, the user issues an FTP command and the server responds with a prompt primary reply. The user should wait for this initial primary success or failure response before sending further commands.

Certain commands require a second reply for which the user should also wait. These replies may, for example, report on the progress or completion of file transfer or the closing of the data connection. They are secondary replies to file transfer commands.

One important group of informational replies is the connection greetings. Under normal circumstances, a server will send a 220 reply, "awaiting input", when the connection is completed. The user should wait for this greeting message before sending any commands. If the server is unable to accept input right away, he should send a 120 "expected delay" reply immediately and a 220 reply when ready. The user will then know not to hang up if there is a delay.

The table below lists alternative success and failure replies for each command. These must be strictly adhered to; a server may substitute text in the replies, but the meaning and action implied by the code numbers and by the specific command reply sequence cannot be altered.

Command-Reply Sequences

In this section, the command-reply sequence is presented. Each command is listed with its possible replies; command groups are listed together. Preliminary replies are listed first (with their succeeding replies indented and under them), then positive and negative completion, and finally intermediary replies with the remaining commands from the sequence following. This listing forms the basis for the state diagrams, which will be presented separately.

Connection Establishment
120
220
220
421



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 247
Keysight, EX-1045
Page 247

```
Login
   USER
      230
      530
      500, 501, 421
      331, 332
   PASS
      230
      202
      530
      500, 501, 503, 421
      332
   ACCT
      230
      202
      530
      500, 501, 503, 421
Logout
   QUIT
      221
      500
   REIN
      120
         220
      220
      421
      500, 502
Transfer parameters
   PORT
      200
      500, 501, 421, 530
   PASV
      227
      500, 501, 502, 421, 530
   MODE, TYPE, STRU
      200
      500, 501, 504, 421, 530
File action commands
   ALLO
      200
      202
      500, 501, 504, 421, 530
   REST
      500, 501, 502, 421, 530
      350
```



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 248
Keysight, EX-1045
Page 248

```
STOR
   125, 150
      (110)
      226, 250
      425, 426, 451, 551, 552
   532, 450, 452, 553
   500, 501, 421, 530
RETR
   125, 150
      (110)
      226, 250
      425, 426, 451
   450, 550
   500, 501, 421, 530
LIST, NLST
   125, 150
      226, 250
      425, 426, 451
   450
   500, 501, 502, 421, 530
APPE
   125, 150
      (110)
      226, 250
      425, 426, 451, 551, 552
   532, 450, 550, 452, 553
   500, 501, 502, 421, 530
MLFL
   125, 150, 151, 152
      226, 250
      425, 426, 451, 552
   532, 450, 550, 452, 553
   500, 501, 502, 421, 530
RNFR
   450, 550
   500, 501, 502, 421, 530
   350
RNTO
   250
   532, 553
   500, 501, 502, 503, 421, 530
DELE, CWD
   250
   450, 550
   500, 501, 502, 421, 530
```



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 249
Keysight, EX-1045
Page 249

```
      ABOR
         225, 226
         500, 501, 502, 421
      MAIL, MSND
         151, 152
            354
               250
               451, 552
         354
            250
            451, 552
         450, 550, 452, 553
         500, 501, 502, 421, 530
      MSOM, MSAM
         119, 151, 152
            354
               250
               451, 552
         354
            250
            451, 552
         450, 550, 452, 553
         500, 501, 502, 421, 530
      MRSQ
         200, 215
         500, 501, 502, 421, 530
      MRCP
         151, 152
            200
         200
         450, 550, 452, 553
         500, 501, 502, 503, 421
   Informational commands
      STAT
         211, 212, 213
         450
         500, 501, 502, 421, 530
      HELP
         211, 214
         500, 501, 502, 421
   Miscellaneous commands
      SITE
         200
         202
         500, 501, 530
```



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 250
Keysight, EX-1045
Page 250

IEN 149 June 1980
RFC 765 File Transfer Protocol

NOOP
200
500 421



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 251
Keysight, EX-1045
Page 251

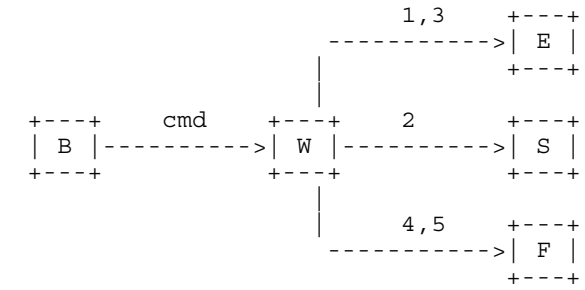
STATE DIAGRAMS

Here we present state diagrams for a very simple minded FTP implementation. Only the first digit of the reply codes is used. There is one state diagram for each group of FTP commands or command sequences.

The command groupings were determined by constructing a model for each command then collecting together the commands with structurally identical models.

For each command or command sequence there are three possible outcomes: success (S), failure (F), and error (E). In the state diagrams below we use the symbol B for "begin", and the symbol W for "wait for reply".

We first present the diagram that represents the largest group of FTP commands:




This diagram models the commands:

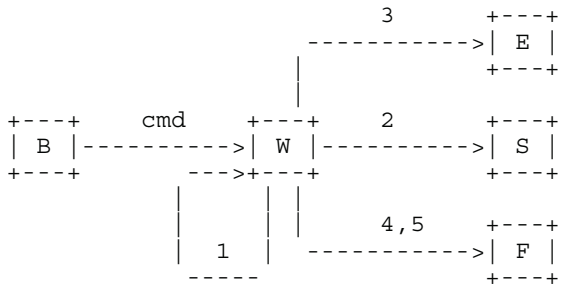
ABOR, ALLO, DELE, CWD, HELP, MODE, MRCP, MRSQ, NOOP, PASV, QUIT, SITE, PORT, STAT, STRU, TYPE.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 252
Keysight, EX-1045
Page 252

The other large group of commands is represented by a very similar diagram:

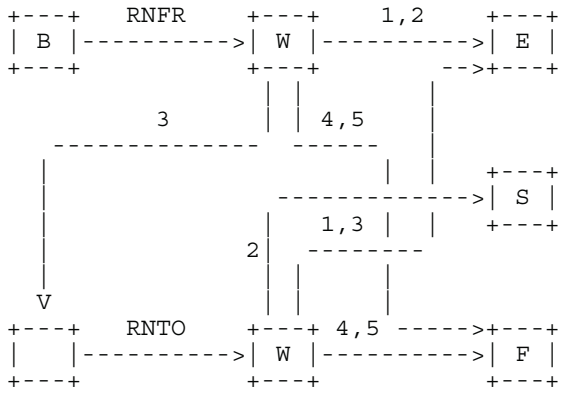



This diagram models the commands:

APPE, LIST, MLFL, NLST, REIN, RETR, STOR.

Note that this second model could also be used to represent the first group of commands, the only difference being that in the first group the 100 series replies are unexpected and therefore treated as error, while the second group expects (some may require) 100 series replies.

The remaining diagrams model command sequences, perhaps the simplest of these is the rename sequence:

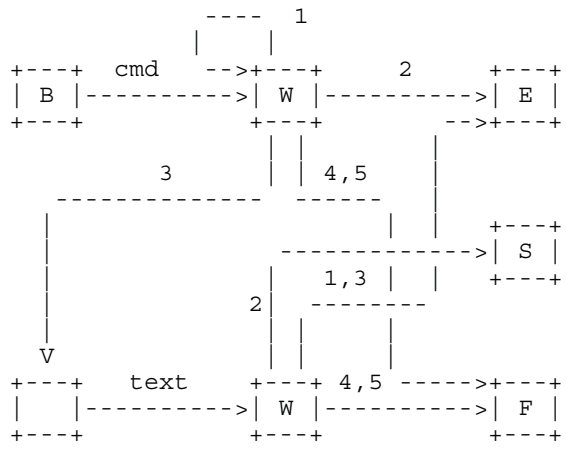





https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 253
Keysight, EX-1045
Page 253

A very similar diagram models the Mail and Send commands:




This diagram models the commands:

MAIL, MSND, MSOM, MSAM.

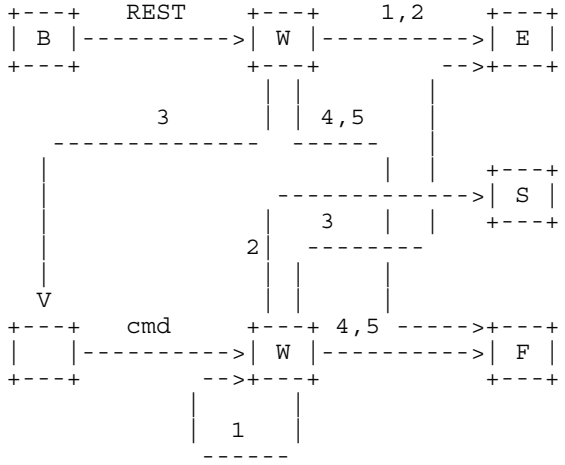
Note that the "text" here is a series of lines sent from the user to the server with no response expected until the last line is sent, recall that the last line must consist only of a single period.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 254
Keysight, EX-1045
Page 254

The next diagram is a simple model of the Restart command:




Where "cmd" is APPE, STOR, RETR, or MLFL.

We note that the above three models are similar, in fact the Mail diagram and the Rename diagram are structurally identical. The Restart differs from the other two only in the treatment of 100 series replies at the second stage.

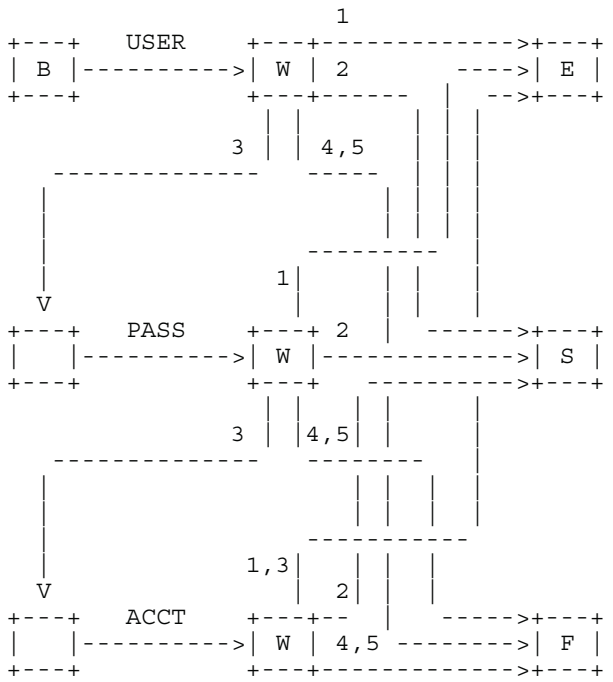


https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 255
Keysight, EX-1045
Page 255

June 1980 IEN 149
File Transfer Protocol RFC 765

The most complicated diagram is for the Login sequence:

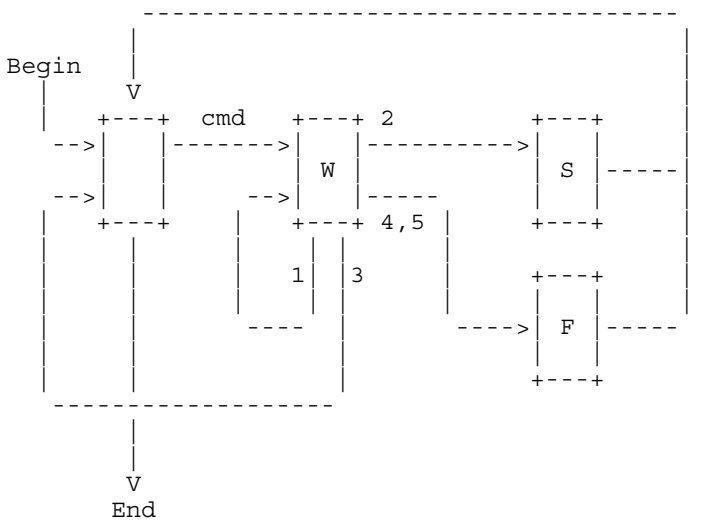





https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 256
Keysight, EX-1045
Page 256

Finally we present a generalized diagram that could be used to model the command and reply interchange:






https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 257
Keysight, EX-1045
Page 257

TYPICAL FTP SCENARIO

User at Host U wanting to transfer files to/from Host S:

In general the user will communicate to the server via a mediating user-FTP process. The following may be a typical scenario. The user-FTP prompts are shown in parentheses, '---->' represents commands from Host U to Host S, and '<----' represents replies from Host S to Host U.

| LOCAL COMMANDS BY USER | ACTION INVOLVED |
|---|---|
| ftp (host) multics<CR> | Connect to Host S, port L, establishing TELNET connections<br><---- 220 Service ready <CRLF> |
| username Doe <CR> | USER Doe<CRLF>----><br><---- 331 User name ok, need password<CRLF> |
| password mumble <CR> | PASS mumble<CRLF>----><br><---- 230 User logged in.<CRLF> |
| retrieve (local type) ASCII<CR><br>(local pathname) test 1 <CR><br>(for.pathname) test.pl1<CR> | User-FTP opens local file in ASCII.<br>RETR test.pl1<CRLF> ----><br><---- 150 File status okay; about to open data connection<br>Server makes data connection to port U |
| <CRLF> | <---- 226 Closing data connection, file transfer successful<CRLF> |
| type Image<CR> | TYPE I<CRLF> ----><br><---- 200 Command OK<CRLF> |
| store (local type) image<CR><br>(local pathname) file dump<CR><br>(for.pathname) >udd>cn>fd<CR> | User-FTP opens local file in Image.<br>STOR >udd>cn>fd<CRLF> ----><br><---- 450 Access denied<CRLF> |
| terminate | QUIT <CRLF> ----><br>Server closes all connections. |



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 258
Keysight, EX-1045
Page 258

IEN 149 June 1980
RFC 765 File Transfer Protocol

CONNECTION ESTABLISHMENT

The FTP control connection is established via TCP between the user process port U and the server process port L. This protocol is assigned the service port 21 (25 octal), that is L=21.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 259
Keysight, EX-1045
Page 259

APPENDIX ON MAIL

The basic commands transmitting mail are the MAIL and the MLFL commands. These commands cause the transmitted data to be entered into the recipients mailbox.

MAIL <SP> <recipient name> <CRLF>

If accepted, returns 354 reply and considers all succeeding lines to be the message text, terminated by a line containing only a period, upon which a 250 completion reply is returned. Various errors are possible.

MLFL <SP> <recipient name> <CRLF>

If accepted, acts like a STOR command, except that the data is considered to be the message text. Various errors are possible.

There are two possible preliminary replies that a server may use to indicate that it is accepting mail for a user whose mailbox is not at that server.

151 User not local; Will forward to <user>@<host>.

This reply indicates that the server knows the user's mailbox is on another host and will take responsibility for forwarding the mail to that host. For example, at BBN (or ISI) there are several host which each have a list of many of the users on several of the host. These hosts then can accept mail for any user on their list and forward it to the correct host.

152 User Unknown; Mail will be forwarded by the operator.

This reply indicates that the host does not recognize the user name, but that it will accept the mail and have the operator attempt to deliver it. This is useful if the user name is misspelled, but may be a disservice if the mail is really undeliverable.

Three FTP commands provide for "sending" a message to a logged-in user's terminal, as well as variants for mailing it normally whether the user is logged in or not.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 260
Keysight, EX-1045
Page 260

MSND -- SeND to terminal.

Returns 450 failure reply if the addressee is refusing or not logged in.

MSOM -- Send to terminal Or Mailbox.

Returns 119 notification reply if terminal is not accessible.

MSAM -- Send to terminal And Mailbox.

Returns 119 notification reply if terminal is not accessible.

Note that for MSOM and MSAM, it is the mailing which determines success, not the sending, although MSOM as implemented uses a 119 reply (in addition to the normal success/failure code) to indicate that because the SEND failed, an attempt is being made to mail the message instead. There are no corresponding variants for MLFL, since messages transmitted in this way are generally short.

There are two FTP commands which allow one to mail the text of a message to several recipients simultaneously; such message transmission is far more efficient than the practice of sending the text again and again for each additional recipient at a site.

There are two basic ways of sending a single text to several recipients. In one, all recipients are specified first, and then the text is sent; in the other, the order is reversed and the text is sent first, followed by the recipients. Both schemes are necessary because neither by itself is optimal for all systems, as will be explained later. To select a particular scheme, the MRSQ command is used; to specify recipients after a scheme is chosen, MRCP commands are given; and to furnish text, the MAIL or MLFL commands are used.

Scheme Selection: MRSQ

MRSQ is the means by which a user program can test for implementation of MRSQ/MRCP, select a particular scheme, reset its state thereof, and even do some rudimentary negotiation. Its format is like that of the TYPE command, as follows:



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 261
Keysight, EX-1045
Page 261

```
      MRSQ [<SP> <scheme>] <CRLF>

      <scheme> = a single character.  The following are defined:
         R  Recipients first.  If not implemented, T must be.
         T  Text first.  If this is not implemented, R must be.
         ?  Request for preference.  Must always be implemented.

         No argument means a "selection" of none of the schemes (the
         default).

      Replies:
         200 OK, we'll use specified scheme.
         215 <scheme> This is the scheme I prefer.
         501 I understand MRSQ but can't use that scheme.
         5xx Command unrecognized or unimplemented.
```

Three aspects of MRSQ need to be pointed out here. The first is that an MRSQ with no argument must always return a 200 reply and restore the default state of having no scheme selected. Any other reply implies that MRSQ and hence MRCP are not understood or cannot be performed correctly.

The second is that the use of "?" as a <scheme> asks the FTP server to return a 215 reply in which the server specifies a "preferred" scheme. The format of this reply is simple:

```
   215 <SP> <scheme> [<SP> <arbitrary text>] <CRLF>
```

Any other reply (e.g. 4xx or 5xx) implies that MRSQ and MRCP are not implemented, because "?" must always be implemented if MRSQ is.

The third important thing about MRSQ is that it always has the side effect of resetting all schemes to their initial state. This reset must be done no matter what the reply will be - 200, 215, or 501. The actions necessary for a reset will be explained when discussing how each scheme actually works.

Message Text Specification: MAIL/MLFL

Regardless of which scheme (if any) has been selected, a MAIL or MLFL with a non-null argument will behave exactly as before; the MRSQ/MRCP commands have no effect on them. However, such normal MAIL/MLFL commands do have the same side effect as MRSQ; they "reset" the current scheme to its initial state.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 262
Keysight, EX-1045
Page 262

It is only when the argument is null (e.g. MAIL<CRLF> or MLFL<CRLF>) that the particular scheme being used is important, because rather than producing an error (as most servers currently do), the server will accept message text for this "null" specification; what it does with it depends on which scheme is in effect, and will be described in "Scheme Mechanics".

Recipient specification: MRCP

In order to specify recipient names (i.e., idents) and receive some acknowledgment (or refusal) for each name, the following command is used:

MRCP <SP> <ident> <CRLF>

Reply for no scheme:
503 No scheme specified yet; use MRSQ.
Replies for scheme T are identical to those for MAIL/MLFL.
Replies for scheme R (recipients first):
200 OK, name stored.
452 Recipient table full, this name not stored.
553 Recipient name rejected.
4xx Temporary error, try this name again later.
5xx Permanent error, report to sender.

Note that use of this command is an error if no scheme has been selected yet; an MRSQ <scheme> must have been given if MRCP is to be used.

Scheme mechanics: MRSQ R (Recipients first)

In the recipients-first scheme, MRCP is used to specify names which the FTP server stores in a list or table. Normally the reply for each MRCP will be either a 200 for acceptance, or a 4xx/5xx code for rejection; all 5xx codes are permanent rejections (e.g. user not known) which should be reported to the human sender, whereas 4xx codes in general connote some temporary error that may be rectified later. None of the 4xx/5xx replies impinge on previous or succeeding MRCP commands, except for 452 which indicates that no further MRCP's will succeed unless a message is sent to the already stored recipients or a reset is done.

Sending message text to stored recipients is done by giving a MAIL or MLFL command with no argument; that is, just MAIL<CRLF> or MLFL<CRLF>. Transmission of the message text is exactly the same as for normal MAIL/MLFL; however, a positive acknowledgment at the



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 263
Keysight, EX-1045
Page 263

end of transmission means that the message has been sent to ALL recipients that were remembered with MRCP, and a failure code means that it should be considered to have failed for ALL of these specified recipients. This applies regardless of the actual error code; and whether the reply signifies success or failure, all stored recipient names are flushed and forgotten - in other words, things are reset to their initial state. This purging of the recipient name list must also be done as the "reset" side effect of any use of MRSQ.

A 452 reply to an MRCP can thus be handled by using a MAIL/MLFL to specify the message for currently stored recipients, and then sending more MRCP's and another MAIL/MLFL, as many times as necessary; for example, if a server only had room for 10 names this would result in a 50-recipient message being sent 5 times, to 10 different recipients each time.

If a user attempts to specify message text (MAIL/MLFL with no argument) before any successful MRCP's have been given, this should be treated exactly as a "normal" MAIL/MLFL with a null recipient would be; some servers will return an error of some type, such as "550 Null recipient".

See Example 1 for an example using MRSQ R.

Scheme mechanics: MRSQ T (Text first)

In the text-first scheme, MAIL/MLFL with no argument is used to specify message text, which the server stores away. Succeeding MRCP's are then treated as if they were MAIL/MLFL commands, except that none of the text transfer manipulations are done; the stored message text is sent to the specified recipient, and a reply code is returned identical to that which an actual MAIL/MLFL would invoke. (Note ANY 2xx code indicates success.)

The stored message text is not forgotten until the next MAIL/MLFL or MRSQ, which will either replace it with new text or flush it entirely. Any use of MRSQ will reset this scheme by flushing stored text, as will any use of MAIL/MLFL with a non-null argument.

If an MRCP is seen before any message text has been stored, the user in effect is trying to send a null message; some servers might allow this, others would return an error code.

See Example 2 for an example using MRSQ T.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 264
Keysight, EX-1045
Page 264

Why two schemes anyway?

Because neither by itself is optimal for all systems. MRSQ R allows more of a "bulk" mailing, because everything is saved up and then mailed simultaneously; this is very useful for systems such as ITS where the FTP server does not itself write mail directly, but hands it on to a central mailer demon of great power; the more information (e.g. recipients) associated with a single "hand-off", the more efficiently mail can be delivered.

By contrast, MRSQ T is geared to FTP servers which want to deliver mail directly, in one-by-one incremental fashion. This way they can return an individual success/failure reply code for each recipient given which may depend on variable file system factors such as exceeding disk allocation, mailbox access conflicts, and so forth; if they tried to emulate MRSQ R's bulk mailing, they would have to ensure that a success reply to the MAIL/MLFL indeed meant that it had been delivered to ALL recipients specified - not just some.

Notes:

* Because these commands are not required in the minimum implementation of FTP, one must be prepared to deal with sites which don't recognize either MRSQ or MRCP. "MRSQ" and "MRSQ ?" are explicitly designed as tests to see whether either scheme is implemented; MRCP is not, and a failure return of the "unimplemented" variety could be confused with "No scheme selected yet", or even with "Recipient unknown". Be safe, be sure, use MRSQ!

* There is no way to indicate in a positive response to "MRSQ ?" that the preferred "scheme" for a server is that of the default state; i.e. none of the multi-recipient schemes. The rationale is that in this case, it would be pointless to implement MRSQ/MRCP at all, and the response would therefore be negative.

* One reason that the use of MAIL/MLFL is restricted to null arguments with this multi-recipient extension is the ambiguity that would result if a non-null argument were allowed; for example, if MRSQ R was in effect and some MRCP's had been given, and a MAIL FOO<CRLF> was done, there would be no way to distinguish a failure reply for mailbox "FOO" from a global failure for all recipients specified. A similar situation exists for MRSQ T; it would not be clear whether the text was stored and the mailbox failed, or vice versa, or both.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 265
Keysight, EX-1045
Page 265

* "Resets" are done by all MRSQ's and "normal" MAIL/MLFL's to avoid confusion and overly complicated implementation. The MRSQ command implies a change or uncertainty of status, and the latter commands would otherwise have to use some independent mechanisms to avoid clobbering the data bases (e.g., message text storage area) used by the T/R schemes. However, once a scheme is selected, it remains "in effect" just as a "TYPE A" remains selected. The recommended way for doing a reset, without changing the current selection, is with "MRSQ ?". Remember that "MRSQ" alone reverts to the no-scheme state.

* It is permissible to intersperse other FTP commands among the MRSQ/MRCP/MAIL sequences.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 266
Keysight, EX-1045
Page 266

Example 1

Example of MRSQ R (Recipients first)

This is an example of how MRSQ R is used; first the user must establish that the server in fact implements MRSQ:

```
U: MRSQ
S: 200 OK, no scheme selected.
```

An MRSQ with a null argument always returns a 200 if implemented, selecting the "scheme" of null, i.e. none of them. If MRSQ were not implemented, a code of 4xx or 5xx would be returned.

```
U: MRSQ R
S: 200 OK, using that scheme
```

All's well; now the recipients can be specified.

```
U: MRCP Foo
S: 200 OK

U: MRCP Raboof
S: 553 Who's that? No such user here.

U: MRCP bar
S: 200 OK
```

Well, two out of three ain't bad. Note that the demise of "Raboof" has no effect on the storage of "Foo" or "bar". Now to furnish the message text, by giving a MAIL or MLFL with no argument:

```
U: MAIL
S: 354 Type mail, ended by <CRLF>.<CRLF>
U: Blah blah blah blah....etc etc etc
U: .
S: 250 Mail sent.
```

The text has now been sent to both "Foo" and "bar".



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 267
Keysight, EX-1045
Page 267

Example 2

Example of MRSQ T (Text first)

Using the same message as the previous example:

```
U: MRSQ ?
S: 215 T Text first, please.
```

MRSQ is indeed implemented, and the server says that it prefers "T", but that needn't stop the user from trying something else:

```
U: MRSQ R
S: 501 Sorry, I really can't do that.
```

It's possible that it could have understood "R" also, but in general it's best to use the "preferred" scheme, since the server knows which is most efficient for its particular site. Anyway:

```
U: MRSQ T
S: 200 OK, using that scheme.
```

Scheme "T" is now selected, and the text must be sent:

```
U: MAIL
S: 354 Type mail, ended by <CRLF>.<CRLF>
U: Blah blah blah blah....etc etc etc
U: .
S: 250 Mail stored.
```

Now recipients can be specified:

```
U: MRCP Foo
S: 250 Stored mail sent.

U: MRCP Raboof
S: 553 Who's that? No such user here.

U: MRCP bar
S: 250 Stored mail sent.
```



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 268
Keysight, EX-1045
Page 268

Again, the text has now been sent to both "Foo" and "bar", and still remains stored. A new message can be sent with another MAIL/MRCP... sequence, but the fastidious or paranoid could chose to do:

```
U: MRSQ ?
S: 215 T Text first, please.
```

Which resets things without altering the scheme in effect.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 269
Keysight, EX-1045
Page 269

APPENDIX ON PAGE STRUCTURE

The need for FTP to support page structure derives principally from the need to support efficient transmission of files between TOPS20 systems, particularly the files used by NLS.

The file system of TOPS20 is based on the concept of pages. The system level is most efficient at manipulating files as pages. System level programs provide an interface to the file system so that many applications view files as sequential streams of characters. However, a few applications use the underlying page structures directly, and some of these create holey files.

A TOPS20 file is just a bunch of words pointed to by a page table. If those words contain CRLF's, fine -- but that doesn't mean "record" to TOPS20.

A TOPS20 disk file consists of four things: a pathname, a page table, a (possibly empty) set of pages, and a set of attributes.

The pathname is specified in the RETR or STOR command. It includes the directory name, file name, file name extension, and version number.

The page table contains up to 2**18 entries. Each entry may be EMPTY, or may point to a page. If it is not empty, there are also some page-specific access bits; not all pages of a file need have the same access protection.

A page is a contiguous set of 512 words of 36 bits each.

The attributes of the file, in the File Descriptor Block (FDB), contain such things as creation time, write time, read time, writer's byte-size, end of file pointer, count of reads and writes, backup system tape numbers, etc.

Note that there is NO requirement that pages in the page table be contiguous. There may be empty page table slots between occupied ones. Also, the end of file pointer is simply a number. There is no requirement that it in fact point at the "last" datum in the file. Ordinary sequential I/O calls in TOPS20 will cause the end of file pointer to be left after the last datum written, but other operations may cause it not to be so, if a particular programming system so requires.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 270
Keysight, EX-1045
Page 270

In fact both of these special cases, "holey" files and end-of-file pointers not at the end of the file, occur with NLS data files.

The TOPS20 paged files can be sent with the FTP transfer parameters: TYPE L 36, STRU P, and MODE S (in fact any mode could be used).

Each page of information has a header. Each header field, which is a logical byte, is a TOPS20 word, since the TYPE is L 36.

The header fields are:

Word 0: Header Length.

The header length is 5.

Word 1: Page Index.

If the data is a disk file page, this is the number of that page in the file's page map. Empty pages (holes) in the file are simply not sent. Note that a hole is NOT the same as a page of zeros.

Word 2: Data Length.

The number of data words in this page, following the header. Thus the total length of the transmission unit is the Header Length plus the Data Length.

Word 3: Page Type.

A code for what type of chunk this is. A data page is type 3, the FDB page is type 2.

Word 4: Page Access Control.

The access bits associated with the page in the file's page map. (This full word quantity is put into AC2 of an SPACS by the program reading from net to disk.)

After the header are Data Length data words. Data Length is currently either 512 for a data page or 21 for an FDB. Trailing zeros in a disk file page may be discarded, making Data Length less than 512 in that case.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 271
Keysight, EX-1045
Page 271

Data transfers are implemented like the layers of an onion: some characters are packaged into a line. Some lines are packaged into a file. The file is broken into other manageable units for transmission. Those units have compression applied to them. The units may be flagged by restart markers. On the other end, the process is reversed.



https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 272
Keysight, EX-1045
Page 272

https://web.archive.org/web/20070206005843/https://tools.ietf.org/html/rfc765



1991

# The Original HTTP as defined in 1991

This document defines the Hypertext Transfer protocol (HTTP) as originally implemented by the World Wide Web initaitive software in the prototype released. This is a subset of the full HTTP protocol, and is known as HTTP 0.9.

No client profile information is transferred with the query. Future HTTP protocols will be back-compatible with this protocol.

This restricted protocol is very simple and may always be used when you do not need the capabilities of the full protocol which is backwards compatible.

The definition of this protocol is in the public domain (see policy ).

The protocol uses the normal internet-style telnet protocol style on a TCP-IP link. The following describes how a client acquires a (hypertext) document from an HTTP server, given an HTTP document address .

## Connection

The client makes a TCP-IP connection to the host using the domain name or IP number , and the port number given in the address.

If the port number is not specified, 80 is always assumed for HTTP.

The server accepts the connection.

Note: HTTP currently runs over TCP, but could run over any connection-oriented service. The interpretation of the protocol below in the case of a sequenced packet service (such as DECnet(TM) or ISO TP4) is that that the request should be one TPDU, but the response may be many.

## Request

The client sends a document request consisting of a line of ASCII characters terminated by a CR LF (carriage return, line feed) pair. A well-behaved server will not require the carriage return character.

This request consists of the word "GET", a space, the document address , omitting the "http:, host and port parts when they are the coordinates just used to make the connection. (If a gateway is being used, then a full document address may be given specifying a different naming scheme).

The document address will consist of a single word (ie no spaces). If any further words are found on the request line, they MUST either be ignored, or else treated according to the full HTTP spec .



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 274
Keysight, EX-1045
Page 274

The search functionality of the protocol lies in the ability of the addressing syntax to describe a search on a named index .

A search should only be requested by a client when the index document itself has been descibed as an index using the ISINDEX tag .

## Response

The response to a simple GET request is a message in hypertext mark-up language ( HTML ). This is a byte stream of ASCII characters.

Lines shall be delimited by an optional carriage return followed by a mandatory line feed chararcter. The client should not assume that the carriage return will be present. Lines may be of any length. Well-behaved servers should retrict line length to 80 characters excluding the CR LF pair.

The format of the message is HTML - that is, a trimmed SGML document. Note that this format allows for menus and hit lists to be returned as hypertext. It also allows for plain ASCII text to be returned following the PLAINTEXT tag .

The message is terminated by the closing of the connection by the server.

Well-behaved clients will read the entire document as fast as possible. The client shall not wait for user action (output paging for example) before reading the whole of the document. The server may impose a timeout of the order of 15 seconds on inactivity.

Error responses are supplied in human readable text in HTML syntax. There is no way to distinguish an error response from a satisfactory response except for the content of the text.

## Disconnection

The TCP-IP connection is broken by the server when the whole document has been transferred.

The client may abort the transfer by breaking the connection before this, in which case the server shall not record any error condition.

Requests are idempotent . The server need not store any information about the request after disconnection.

*Tim BL*

https://web.archive.org/web/19970605071155/http://www.w3.org:80/Protocols/HTTP/AsImplemented.html

Network Working Group
Request for Comments: 2818
Category: Informational

E. Rescorla
RTFM, Inc.

May 2000

# HTTP Over TLS

## Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

## Copyright Notice

Copyright (C) The Internet Society (2000). All Rights Reserved.

## Abstract

This memo describes how to use TLS to secure HTTP connections over the Internet. Current practice is to layer HTTP over SSL (the predecessor to TLS), distinguishing secured traffic from insecure traffic by the use of a different server port. This document documents that practice using TLS. A companion document describes a method for using HTTP/TLS over the same port as normal HTTP [RFC2817].

## Table of Contents

1. Introduction . . . . . 2
1.1. Requirements Terminology . . . . . 2
2. HTTP Over TLS . . . . . 2
2.1. Connection Initiation . . . . . 2
2.2. Connection Closure . . . . . 2
2.2.1. Client Behavior . . . . . 3
2.2.2. Server Behavior . . . . . 3
2.3. Port Number . . . . . 4
2.4. URI Format . . . . . 4
3. Endpoint Identification . . . . . 4
3.1. Server Identity . . . . . 4
3.2. Client Identity . . . . . 5
References . . . . . 6
Security Considerations . . . . . 6
Author's Address . . . . . 6
Full Copyright Statement . . . . . 7



https://web.archive.org/web/20070125003236/https://tools.ietf.org/html/rfc2818

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 276
Keysight, EX-1045
Page 276

## 1. Introduction

HTTP [RFC2616] was originally used in the clear on the Internet. However, increased use of HTTP for sensitive applications has required security measures. SSL, and its successor TLS [RFC2246] were designed to provide channel-oriented security. This document describes how to use HTTP over TLS.

### 1.1. Requirements Terminology

Keywords "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT" and "MAY" that appear in this document are to be interpreted as described in [RFC2119].

## 2. HTTP Over TLS

Conceptually, HTTP/TLS is very simple. Simply use HTTP over TLS precisely as you would use HTTP over TCP.

### 2.1. Connection Initiation

The agent acting as the HTTP client should also act as the TLS client. It should initiate a connection to the server on the appropriate port and then send the TLS ClientHello to begin the TLS handshake. When the TLS handshake has finished. The client may then initiate the first HTTP request. All HTTP data MUST be sent as TLS "application data". Normal HTTP behavior, including retained connections should be followed.

### 2.2. Connection Closure

TLS provides a facility for secure connection closure. When a valid closure alert is received, an implementation can be assured that no further data will be received on that connection. TLS implementations MUST initiate an exchange of closure alerts before closing a connection. A TLS implementation MAY, after sending a closure alert, close the connection without waiting for the peer to send its closure alert, generating an "incomplete close". Note that an implementation which does this MAY choose to reuse the session. This SHOULD only be done when the application knows (typically through detecting HTTP message boundaries) that it has received all the message data that it cares about.

As specified in [RFC2246], any implementation which receives a connection close without first receiving a valid closure alert (a "premature close") MUST NOT reuse that session. Note that a premature close does not call into question the security of the data already received, but simply indicates that subsequent data might



https://web.archive.org/web/20070125003236/https://tools.ietf.org/html/rfc2818

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 277
Keysight, EX-1045
Page 277

have been truncated. Because TLS is oblivious to HTTP request/response boundaries, it is necessary to examine the HTTP data itself (specifically the Content-Length header) to determine whether the truncation occurred inside a message or between messages.

### 2.2.1. Client Behavior

Because HTTP uses connection closure to signal end of server data, client implementations MUST treat any premature closes as errors and the data received as potentially truncated. While in some cases the HTTP protocol allows the client to find out whether truncation took place so that, if it received the complete reply, it may tolerate such errors following the principle to "[be] strict when sending and tolerant when receiving" [RFC1958], often truncation does not show in the HTTP protocol data; two cases in particular deserve special note:

A HTTP response without a Content-Length header. Since data length in this situation is signalled by connection close a premature close generated by the server cannot be distinguished from a spurious close generated by an attacker.

A HTTP response with a valid Content-Length header closed before all data has been read. Because TLS does not provide document oriented protection, it is impossible to determine whether the server has miscomputed the Content-Length or an attacker has truncated the connection.

There is one exception to the above rule. When encountering a premature close, a client SHOULD treat as completed all requests for which it has received as much data as specified in the Content-Length header.

A client detecting an incomplete close SHOULD recover gracefully. It MAY resume a TLS session closed in this fashion.

Clients MUST send a closure alert before closing the connection. Clients which are unprepared to receive any more data MAY choose not to wait for the server's closure alert and simply close the connection, thus generating an incomplete close on the server side.

### 2.2.2. Server Behavior

RFC 2616 permits an HTTP client to close the connection at any time, and requires servers to recover gracefully. In particular, servers SHOULD be prepared to receive an incomplete close from the client, since the client can often determine when the end of server data is. Servers SHOULD be willing to resume TLS sessions closed in this fashion.



https://web.archive.org/web/20070125003236/https://tools.ietf.org/html/rfc2818

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 278
Keysight, EX-1045
Page 278

Implementation note: In HTTP implementations which do not use persistent connections, the server ordinarily expects to be able to signal end of data by closing the connection. When Content-Length is used, however, the client may have already sent the closure alert and dropped the connection.

Servers MUST attempt to initiate an exchange of closure alerts with the client before closing the connection. Servers MAY close the connection after sending the closure alert, thus generating an incomplete close on the client side.

### 2.3. Port Number

The first data that an HTTP server expects to receive from the client is the Request-Line production. The first data that a TLS server (and hence an HTTP/TLS server) expects to receive is the ClientHello. Consequently, common practice has been to run HTTP/TLS over a separate port in order to distinguish which protocol is being used. When HTTP/TLS is being run over a TCP/IP connection, the default port is 443. This does not preclude HTTP/TLS from being run over another transport. TLS only presumes a reliable connection-oriented data stream.

### 2.4. URI Format

HTTP/TLS is differentiated from HTTP URIs by using the 'https' protocol identifier in place of the 'http' protocol identifier. An example URI specifying HTTP/TLS is:

```
https://www.example.com/~smith/home.html
```

## 3. Endpoint Identification

### 3.1. Server Identity

In general, HTTP/TLS requests are generated by dereferencing a URI. As a consequence, the hostname for the server is known to the client. If the hostname is available, the client MUST check it against the server's identity as presented in the server's Certificate message, in order to prevent man-in-the-middle attacks.

If the client has external information as to the expected identity of the server, the hostname check MAY be omitted. (For instance, a client may be connecting to a machine whose address and hostname are dynamic but the client knows the certificate that the server will present.) In such cases, it is important to narrow the scope of acceptable certificates as much as possible in order to prevent man



https://web.archive.org/web/20070125003236/https://tools.ietf.org/html/rfc2818

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 279
Keysight, EX-1045
Page 279

in the middle attacks. In special cases, it may be appropriate for the client to simply ignore the server's identity, but it must be understood that this leaves the connection open to active attack.

If a subjectAltName extension of type dNSName is present, that MUST be used as the identity. Otherwise, the (most specific) Common Name field in the Subject field of the certificate MUST be used. Although the use of the Common Name is existing practice, it is deprecated and Certification Authorities are encouraged to use the dNSName instead.

Matching is performed using the matching rules specified by [RFC2459]. If more than one identity of a given type is present in the certificate (e.g., more than one dNSName name, a match in any one of the set is considered acceptable.) Names may contain the wildcard character * which is considered to match any single domain name component or component fragment. E.g., *.a.com matches foo.a.com but not bar.foo.a.com. f*.com matches foo.com but not bar.com.

In some cases, the URI is specified as an IP address rather than a hostname. In this case, the iPAddress subjectAltName must be present in the certificate and must exactly match the IP in the URI.

If the hostname does not match the identity in the certificate, user oriented clients MUST either notify the user (clients MAY give the user the opportunity to continue with the connection in any case) or terminate the connection with a bad certificate error. Automated clients MUST log the error to an appropriate audit log (if available) and SHOULD terminate the connection (with a bad certificate error). Automated clients MAY provide a configuration setting that disables this check, but MUST provide a setting which enables it.

Note that in many cases the URI itself comes from an untrusted source. The above-described check provides no protection against attacks where this source is compromised. For example, if the URI was obtained by clicking on an HTML page which was itself obtained without using HTTP/TLS, a man in the middle could have replaced the URI. In order to prevent this form of attack, users should carefully examine the certificate presented by the server to determine if it meets their expectations.

### 3.2. Client Identity

Typically, the server has no external knowledge of what the client's identity ought to be and so checks (other than that the client has a certificate chain rooted in an appropriate CA) are not possible. If a server has such knowledge (typically from some source external to HTTP or TLS) it SHOULD check the identity as described above.



https://web.archive.org/web/20070125003236/https://tools.ietf.org/html/rfc2818

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 280
Keysight, EX-1045
Page 280

## References

[RFC2459] Housley, R., Ford, W., Polk, W. and D. Solo, "Internet Public Key Infrastructure: Part I: X.509 Certificate and CRL Profile", RFC 2459, January 1999.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. and T. Berners-Lee, "Hypertext Transfer Protocol, HTTP/1.1", RFC 2616, June 1999.

[RFC2119] Bradner, S., "Key Words for use in RFCs to indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2246] Dierks, T. and C. Allen, "The TLS Protocol", RFC 2246, January 1999.

[RFC2817] Khare, R. and S. Lawrence, "Upgrading to TLS Within HTTP/1.1", RFC 2817, May 2000.

## Security Considerations

This entire document is about security.

## Author's Address

Eric Rescorla
RTFM, Inc.
30 Newell Road, #16
East Palo Alto, CA 94303

Phone: (650) 328-8631
EMail: ekr@rtfm.com



https://web.archive.org/web/20070125003236/https://tools.ietf.org/html/rfc2818

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 281
Keysight, EX-1045
Page 281

## Full Copyright Statement

Copyright (C) The Internet Society (2000). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.



https://web.archive.org/web/20070125003236/https://tools.ietf.org/html/rfc2818

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 282
**Keysight, EX-1045**
**Page 282**

https://web.archive.org/web/20070125003236/https://tools.ietf.org/html/rfc2818

RECHERCHER LE BLOG SIGNALER LE BLOG Blog suivant»Créer un blog | Connexion

# MOSERWARE

JEFF MOSER'S SOFTWARE DEVELOPMENT ADVENTURES.

WEDNESDAY, JUNE 10, 2009

## The First Few Milliseconds of an HTTPS Connection

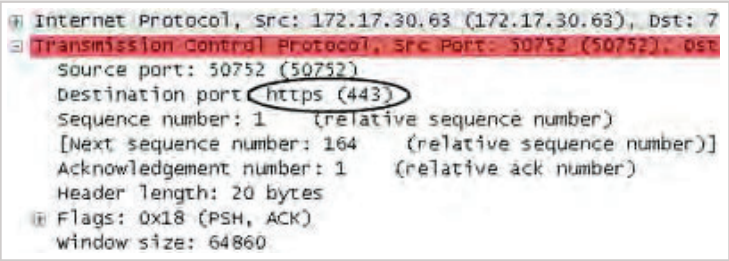
Convinced from spending hours reading rave reviews, Bob eagerly clicked "Proceed to Checkout" for his gallon of Tuscan Whole Milk and...

Whoa! What just happened?




In the 220 milliseconds that flew by, a lot of interesting stuff happened to make Firefox change the address bar color and put a lock in the lower right corner. With the help of Wireshark, my favorite network tool, and a slightly modified debug build of Firefox, we can see *exactly* what's going on.

By agreement of RFC 2818, Firefox knew that "https" meant it should connect to port 443 at Amazon.com:




Most people associate HTTPS with SSL (Secure Sockets Layer) which was created by Netscape in the mid 90's. This is becoming less true over time. As Netscape lost market share, SSL's maintenance moved to the Internet Engineering Task Force (IETF). The first post-Netscape version was re-branded as Transport Layer Security (TLS) 1.0 which was released in January 1999. It's rare to see true "SSL" traffic given that TLS has been around for 10 years.

### Client Hello

TLS wraps all traffic in "records" of different types. We see that the first byte

ABOUT ME



JEFF MOSER
FISHERS, INDIANA, UNITED STATES

VIEW MY COMPLETE PROFILE

BLOG ARCHIVE

▼ 2009 (5)
▼ June (1)
The First Few Milliseconds of an HTTPS Connection
► April (1)
► March (1)
► February (1)
► January (1)
► 2008 (20)
► 2007 (15)

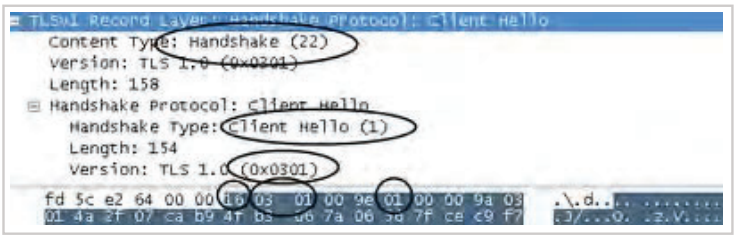
Subscribe
Subscribe by Email
2554 readers BY FEEDBURNER
DEL.ICIO.US
odiogo

CONTACT ME



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 284
Keysight, EX-1045
Page 284

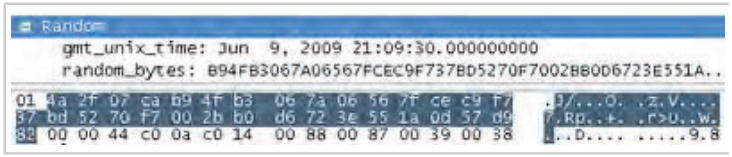
out of our browser is the hex byte 0x16 = 22 which means that this is a "handshake" record:




The next two bytes are 0x0301 which indicate that this is a version 3.1 record which shows that TLS 1.0 is essentially SSL 3.1.

The handshake record is broken out into several messages. The first is our "Client Hello" message (0x01). There are a few important things here:

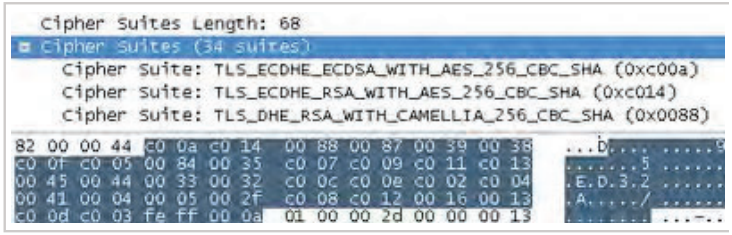
- Random:




  There are four bytes representing the current Coordinated Universal Time (UTC) in the Unix epoch format, which is the number of seconds since January 1, 1970. In this case, 0x4a2f07ca. It's followed by 28 random bytes. This will be used later on.
- Session ID:




  Here it's empty/null. If we had previously connected to Amazon.com a few seconds ago, we could potentially resume a session and avoid a full handshake.
- Cipher Suites:




  This is a list of all of the encryption algorithms that the browser is willing to support. Its top pick is a very strong choice of "TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA" followed by 33 others that it's willing to accept. Don't worry if none of that makes sense. We'll find out later that Amazon doesn't pick our first choice anyway.
- server_name extension:

jeff@moserware.com

Chat with Jeff Moser
Offline

MY RECENT RSS FINDS

CLR via C#, 3rd Edition

The Best Bits

Flashback - A Story of Flash Memory

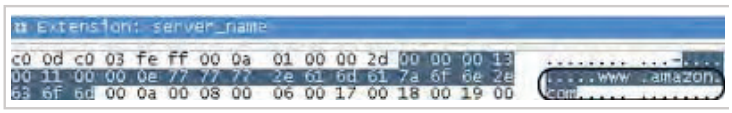
What does the optimize switch do?

A Working Theory About RC4

Read more...



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 285
Keysight, EX-1045
Page 285

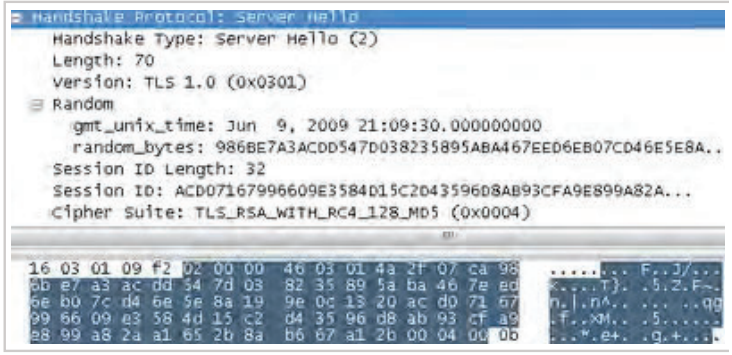



This is a way to tell Amazon.com that our browser is trying to reach https://www.amazon.com/. This is really convenient because our TLS handshake occurs long before any HTTP traffic. HTTP has a "Host" header which allows a cost-cutting Internet hosting companies to pile hundreds of websites onto a single IP address. SSL has traditionally required a different IP for each site, but this extension allows the server to respond with the appropriate certificate that the browser is looking for. If nothing else, this extension should allow an extra week or so of IPv4 addresses.

### Server Hello

Amazon.com replies with a handshake record that's a massive two packets in size (2,551 bytes). The record has version bytes of 0x0301 meaning that Amazon agreed to our request to use TLS 1.0. This record has three sub-messages with some interesting data:

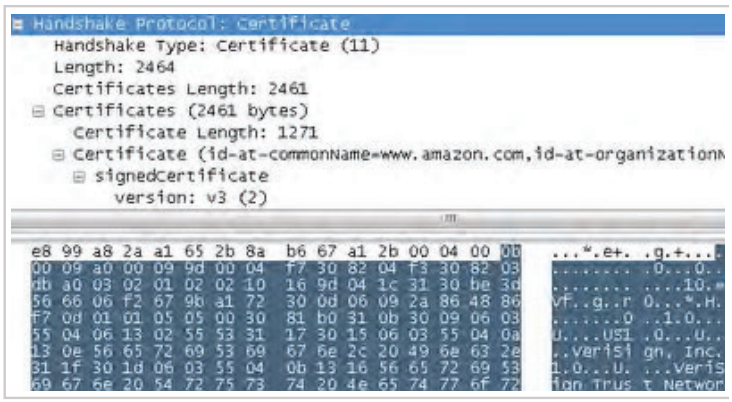
1. "Server Hello" Message (2):



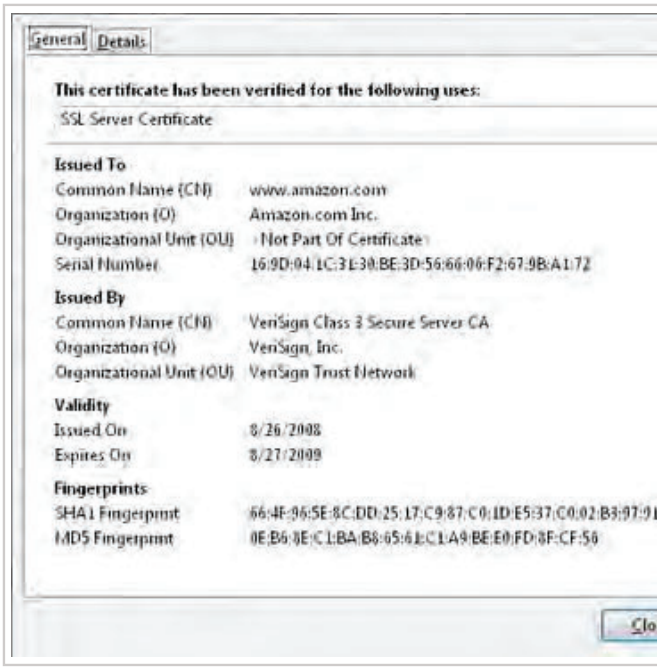

   - We get the server's four byte time Unix epoch time representation and its 28 random bytes that will be used later.
   - A 32 byte session ID in case we want to reconnect without a big handshake.
   - Of the 34 cipher suites we offered, Amazon picked "TLS_RSA_WITH_RC4_128_MD5" (0x0004). This means that it will use the "RSA" public key algorithm to verify certificate signatures and exchange keys, the RC4 encryption algorithm to encrypt data, and the MD5 hash function to verify the contents of messages. We'll cover these in depth later on. I personally think Amazon had selfish reasons for choosing this cipher suite. Of the ones on the list, it was the one that was least CPU intensive to use so that Amazon could crowd more connections onto each of their servers. A much less likely possibility is that they wanted to pay special tribute to Ron Rivest, who created all three of these algorithms.
2. Certificate Message (11):



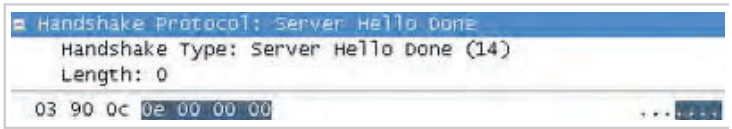
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 286
Keysight, EX-1045
Page 286




- This message takes a whopping 2,464 bytes and is the certificate that the client can use to validate Amazon's. It isn't anything fancy. You can view most of its contents in your browser:




3. "Server Hello Done" Message (14):




- This is a zero byte message that tells the client that it's done with the "Hello" process and indicate that the server won't be asking the client for a certificate.

### Checking out the Certificate

The browser has to figure out if it should trust Amazon.com. In this case, it's



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 287
Keysight, EX-1045
Page 287

using certificates. It looks at Amazon's certificate and sees that the current time is between the "not before" time of August 26th, 2008 and before the "not after" time of August 27, 2009. It also checks to make sure that the certificate's public key is authorized for exchanging secret keys.

Why should we trust this certificate?

Attached to the certificate is a "signature" that is just a really long number in big-endian format:




Anyone could have sent us these bytes. Why should we trust this signature? To answer that question, need to make a speedy detour into mathemagic land:

**Interlude: A Short, Not *Too* Scary, Guide to RSA**

People sometimes wonder if math has any relevance to programming. Certificates give a very practical example of applied math. Amazon's certificate tells us that we should use the RSA algorithm to check the signature. RSA was created in the 1970's by MIT professors Ron *R*ivest, Adi *S*hamir, and Len *A*dleman who found a clever way to combine ideas spanning 2000 years of math development to come up with a beautifully simple algorithm:

You pick two huge prime numbers "p" and "q." Multiply them to get "n = p*q." Next, you pick a small public exponent "e" which is the "encryption exponent" and a specially crafted inverse of "e" called "d" as the "decryption exponent." You then **make "n" and "e" public and keep "d" as secret as you possibly can** and then throw away "p" and "q" (or keep them as secret as "d"). It's really important to remember that "e" and "d" are inverses of each other.

Now, if you have some message, you just need to interpret its bytes as a number "M." If you want to "encrypt" a message to create a "ciphertext", you'd calculate:

$C \equiv M^e \pmod{n}$

This means that you multiply "M" by itself "e" times. The "mod n" means that we only take the remainder (e.g. "modulus") when dividing by "n." For example, 11 AM + 3 hours ≡ 2 (PM) (mod 12 hours). The other recipient knows "d" which allows them to invert the message to recover the original message:

$C^d \equiv (M^e)^d \equiv M^{e*d} \equiv M^1 \equiv M \pmod{n}$



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 288
Keysight, EX-1045
Page 288

Just as interesting is that the person with "d" can "sign" a document by raising a message "M" to the "d" exponent:

$M^d \equiv S \pmod n$

This works because "signer" makes public "S", "M", "e", and "n." Anyone can verify the signature "S" with a simple calculation:

$S^e \equiv (M^d)^e \equiv M^{d*e} \equiv M^{e*d} \equiv M^1 \equiv M \pmod n$

Public key cryptography algorithms like RSA are often called "asymmetric" algorithms because the encryption key (in our case, "e") is not equal to (e.g. "symmetric" with) the decryption key "d". Reducing everything "mod n" makes it impossible to use the easy techniques that we're used to such as normal logarithms. The magic of RSA works because you can calculate/encrypt $C \equiv M^e \pmod n$ very quickly, but it is *really hard* to calculate/decrypt $C^d \equiv M \pmod n$ without knowing "d." As we saw earlier, "d" is derived from factoring "n" back to its "p" and "q", which is a tough problem.

**Verifying Signatures**

The big thing to keep in mind with RSA in the real world is that all of the numbers involved have to be *big* to make things really hard to break using the best algorithms that we have. How big? Amazon.com's certificate was "signed" by "VeriSign Class 3 Secure Server CA." From the certificate, we see that this VeriSign modulus "n" is 2048 bits long which has this 617 digit base-10 representation:

> 1890572922 9464742433 9498401781 6528521078 8629616064
> 3051642608 4317020197 7241822595 6075980039 8371048211
> 4887504542 4200635317 0422636532 2091550579 0341204005
> 1169453804 7325464426 0479594122 4167270607 6731441028
> 3698615569 9947933786 3789783838 5829991518 1037601365
> 0218058341 7944190228 0926880299 3425241541 4300090021
> 1055372661 2125414429 9349272172 5333752665 6605550620
> 5558450610 3253786958 8361121949 2417723618 5199653627
> 5260212221 0847786057 9342235500 9443918198 9038906234
> 1550747726 8041766919 1500918876 1961879460 3091993360
> 6376719337 6644159792 1249204891 7079005527 7689341573
> 9395596650 5484628101 0469658502 1566385762 0175231997
> 6268718746 7514321

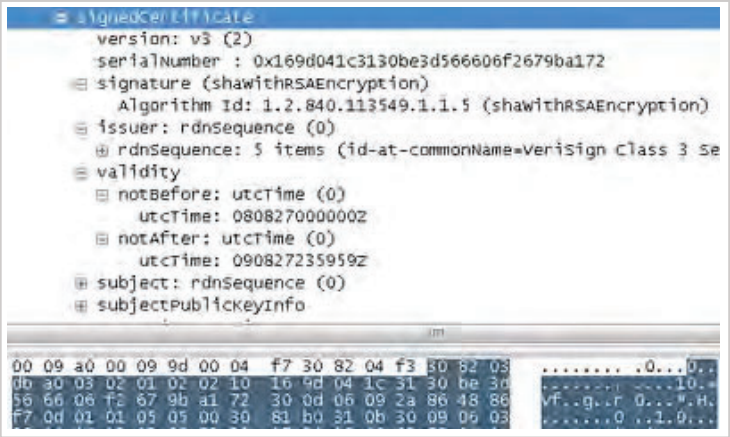
(Good luck trying to find "p" and "q" from this "n" - if you could, you could generate real-looking VeriSign certificates.)

VeriSign's "e" is 2^16 + 1 = 65537. Of course, they keep their "d" value secret, probably on a safe hardware device protected by retinal scanners and armed guards. Before signing, VeriSign checked the validity of the contents that Amazon.com claimed on its certificate using a real-world "handshake" that involved looking at several of their business documents. Once VeriSign was satisfied with the documents, they used the SHA-1 hash algorithm to get a



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 289
Keysight, EX-1045
Page 289

hash value of the certificate that had all the claims. In Wireshark, the full certificate shows up as the "signedCertificate" part:




It's sort of a misnomer since it actually means that those are the bytes that the signer is *going to sign* and not the bytes that already include a signature.




The actual signature, "S", is simply called "encrypted" in Wireshark. If we raise "S" to VeriSign's public "e" exponent of 65537 and then take the remainder when divided by the modulus "n", we get this "decrypted" signature hex value:

> 0001FFFFFFFFFFFF FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF
> FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF
> FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF
> FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF
> FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF
> FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF
> FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF
> FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF
> FFFFFFFF00302130 0906052B0E03021A 05000414C19F8786
> 871775C60EFE0542 E4C2167C830539DB

Per the PKCS #1 v1.5 standard, the first byte is "00" and it "ensures that the encryption block, [when] converted to an integer, is less than the modulus." The second byte of "01" indicates that this is a private key operation (e.g. it's a signature). This is followed by a lot of "FF" bytes that are used to pad the result to make sure that it's big enough. The padding is terminated by a "00"



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 290
Keysight, EX-1045
Page 290

byte. It's followed by "30 21 30 09 06 05 2B 0E 03 02 1A 05 00 04 14" which is the PKCS #1 v2.1 way of specifying the SHA-1 hash algorithm. The last 20 bytes are SHA-1 hash digest of the bytes in "signedCertificate."

Since the decrypted value is properly formatted and the last bytes are the same hash value that we can calculate independently, we can assume that whoever knew "VeriSign Class 3 Secure Server CA"'s private key "signed" it. We implicitly trust that only VeriSign knows the private key "d."

We can repeat the process to verify that "VeriSign Class 3 Secure Server CA"'s certificate was signed by VeriSign's "Class 3 Public Primary Certification Authority."

But why should we trust *that*? There are no more levels on the trust chain.




The top "VeriSign Class 3 Public Primary Certification Authority" was signed by *itself*. This certificate has been built into Mozilla products as an implicitly trusted good certificate since version 1.4 of certdata.txt in the Network Security Services (NSS) library. It was checked-in on September 6, 2000 by Netscape's Robert Relyea with the following comment:

> "Make the framework compile with the rest of NSS. Include a 'live' certdata.txt with those certs we have permission to push to open source (additional certs will be added as we get permission from the owners)."

This decision has had a relatively long impact since the certificate has a validity range of January 28, 1996 - August 1, 2028.

As Ken Thompson explained so well in his "Reflections on Trusting Trust", you ultimately have to implicitly trust somebody. There is no way around this problem. In this case, we're implicitly trusting that Robert Relyea made a good choice. We also hope that Mozilla's built-in certificate policy is reasonable for the other built-in certificates.

One thing to keep in mind here is that all these certificates and signatures were simply used to form a trust chain. On the public Internet, VeriSign's root certificate is implicitly trusted by Firefox long before you go to any website. In a company, you can create your own root certificate authority (CA) that you can install on everyone's machine.

Alternatively, you can get around having to pay companies like VeriSign and



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 291
Keysight, EX-1045
Page 291

avoid certificate trust chains altogether. Certificates are used to establish trust by using a trusted third-party (in this case, VeriSign). If you have a secure means of sharing a secret "key", such as whispering a long password into someone's ear, then you can use that pre-shared key (PSK) to establish trust. There are extensions to TLS to allow this, such as TLS-PSK, and my personal favorite, TLS with Secure Remote Password (SRP) extensions. Unfortunately, these extensions aren't nearly as widely deployed and supported, so they're usually not practical. Additionally, these alternatives impose a burden that we have to have some other secure means of communicating the secret that's more cumbersome than what we're trying to establish with TLS (otherwise, why wouldn't we use *that* for everything?).

One final check that we need to do is to verify that the host name on the certificate is what we expected. Nelson Bolyard's comment in the SSL_AuthCertificate function explains why:

```
/* cert is OK. This is the client side of an SSL connection
 * Now check the name field in the cert against the desired
 * NB: This is our only defense against Man-In-The-Middle (M
```

This check helps prevent against a man-in-the-middle attack because we are implicitly trusting that the people on the certificate trust chain wouldn't do something bad, like sign a certificate claiming to be from Amazon.com unless it actually was Amazon.com. If an attacker is able to modify your DNS server by using a technique like DNS cache poisoning, you might be fooled into thinking you're at a trusted site (like Amazon.com) because the address bar will look normal. This last check implicitly trusts certificate authorities to stop these bad things from happening.

**Pre-Master Secret**

We've verified some claims about Amazon.com and know its public encryption exponent "e" and modulus "n." Anyone listening in on the traffic can know this as well (as evidenced because we are using Wireshark captures). Now we need to create a random secret key that an eavesdropper/attacker can't figure out. This isn't as easy as it sounds. In 1996, researchers figured out that Netscape Navigator 1.1 was using only three sources to seed their pseudo-random number generator (PRNG). The sources were: the time of day, the process id, and the parent process id. As the researchers showed, these "random" sources aren't that random and were relatively easy to figure out.

Since everything else was derived from these three "random" sources, it was possible to "break" the SSL "security" in 25 seconds on a 1996 era machine. If you still don't believe that finding randomness is hard, just ask the Debian OpenSSL maintainers. If you mess it up, all the security built on top of it is suspect.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 292
Keysight, EX-1045
Page 292

On Windows, random numbers used for cryptographic purposes are generated by calling the CryptGenRandom function that hashes bits sampled from over 125 sources. Firefox uses this function along with some bits derived from its own function to seed its pseudo-random number generator.

The 48 byte "pre-master secret" random value that's generated isn't used directly, but it's very important to keep it secret since a lot of things are derived from it. Not surprisingly, Firefox makes it hard to find out this value. I had to compile a debug version and set the SSLDEBUGFILE and SSLTRACE environment variables to see it.

In this particular session, the pre-master secret showed up in the SSLDEBUGFILE as:

```
4456: SSL[131491792]: Pre-Master Secret [Len: 48]
03 01 bb 7b 08 98 a7 49 de e8 e9 b8 91 52 ec 81 ...{...I.....R..
4c c2 39 7b f6 ba 1c 0a b1 95 50 29 be 02 ad e6 L.9{......P)....
ad 6e 11 3f 20 c4 66 f0 64 22 57 7e e1 06 7a 3b .n.? .f.d"W~..z;
```

Note that it's not completely random. The first two bytes are, by convention, the TLS version (03 01).

**Trading Secrets**

We now need to get this secret value over to Amazon.com. By Amazon's wishes of "TLS_RSA_WITH_RC4_128_MD5", we will use RSA to do this. You *could* make your input message equal to just the 48 byte pre-master secret, but the Public Key Cryptography Standard (PKCS) #1, version 1.5 RFC tells us that we should pad these bytes with *random* data to make the input equal to exactly the size of the modulus (1024 bits/128 bytes). This makes it harder for an attacker to determine our pre-master secret. It also gives us one last chance to protect ourselves in case we did something really bone-headed, like reusing the same secret. If we reused the key, the eavesdropper would likely see a different value placed on the network due to the random padding.

Again, Firefox makes it hard to see these random values. I had to insert debugging statements into the padding function to see what was going on:

```
wrapperHandle = fopen("plaintextpadding.txt", "a");
fprintf(wrapperHandle, "PLAINTEXT = ");
for(i = 0; i < modulusLen; i++)
{
    fprintf(wrapperHandle, "%02X ", block[i]);
}
fprintf(wrapperHandle, "\r\n");
fclose(wrapperHandle);
```

In this session, the full padded value was:

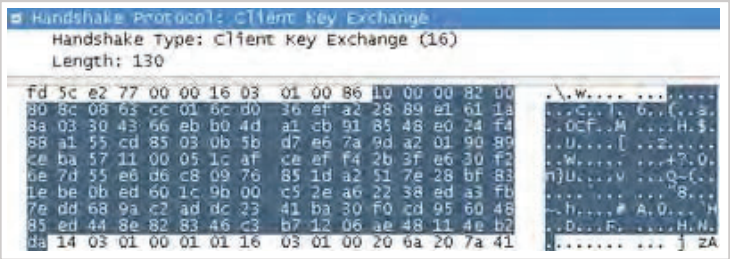
```
00 02 12 A3 EA B1 65 D6 81 6C 13 14 13 62 10 53 23 B3 96 85 FF 24 FA
CC 46 11 21 24 A4 81 EA 30 63 95 D4 DC BF 9C CC D0 2E DD 5A A6
41 6A 4E 82 65 7D 70 7D 50 09 17 CD 10 55 97 B9 C1 A1 84 F2 A9 AB
```



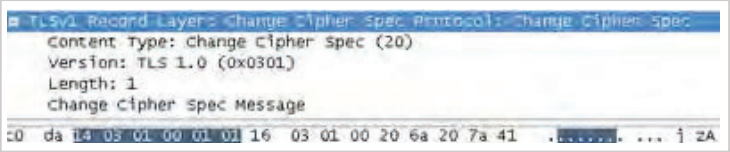
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 293
Keysight, EX-1045
Page 293

> EA 7D F4 CC 54 E4 64 6E 3A E5 91 A0 06 00 03 01 BB 7B 08 98 A7 49
> DE E8 E9 B8 91 52 EC 81 4C C2 39 7B F6 BA 1C 0A B1 95 50 29 BE 02
> AD E6 AD 6E 11 3F 20 C4 66 F0 64 22 57 7E E1 06 7A 3B

Firefox took this value and calculated "$C \equiv M^e \pmod n$" to get the value we see in the "Client Key Exchange" record:




Finally, Firefox sent out one last unencrypted message, a "Change Cipher Spec" record:




This is Firefox's way of telling Amazon that it's going to start using the agreed upon secret to encrypt its next message.

**Deriving the Master Secret**

If we've done everything correctly, both sides (and only those sides) now know the 48 byte (256 bit) pre-master secret. There's a slight trust issue here from Amazon's perspective: the pre-master secret just has bits that were generated by the client, they don't take anything into account from the server or anything we said earlier. We'll fix that be computing the "master secret." Per the spec, this is done by calculating:

> master_secret = PRF(pre_master_secret, "master secret",
> ClientHello.random + ServerHello.random)

The "pre_master_secret" is the secret value we sent earlier. The "master secret" is simply a string whose ASCII bytes (e.g. "6d 61 73 74 65 72 ...") are used. We then concatenate the random values that were sent in the ClientHello and ServerHello (from Amazon) messages that we saw at the beginning.

The PRF is the "Pseudo-Random Function" that's also defined in the spec and is quite clever. It combines the secret, the ASCII label, and the seed data we give it by using the keyed-Hash Message Authentication Code (HMAC) versions of both MD5 and SHA-1 hash functions. Half of the input is sent to each hash function. It's clever because it is quite resistant to attack, even in



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 294
Keysight, EX-1045
Page 294

the face of weaknesses in MD5 and SHA-1. This process can feedback on itself and iterate forever to generate as many bytes as we need.

Following this procedure, we obtain a 48 byte "master secret" of

> 4C AF 20 30 8F 4C AA C5 66 4A 02 90 F2 AC 10 00 39 DB 1D E0 1F CB E0 E0 9D D7 E6 BE 62 A4 6C 18 06 AD 79 21 DB 82 1D 53 84 DB 35 A7 1F C1 01 19

**Generating Lots of Keys**

Now that both sides have a "master secrets", the spec shows us how we can derive all the needed session keys we need using the PRF to create a "key block" where we will pull data from:

> key_block = PRF(SecurityParameters.master_secret, "key expansion", SecurityParameters.server_random + SecurityParameters.client_random);

The bytes from "key_block" are used to populate the following:

> client_write_MAC_secret[SecurityParameters.hash_size]
> server_write_MAC_secret[SecurityParameters.hash_size]
> client_write_key[SecurityParameters.key_material_length]
> server_write_key[SecurityParameters.key_material_length]
> client_write_IV[SecurityParameters.IV_size]
> server_write_IV[SecurityParameters.IV_size]

Since we're using a stream cipher instead of a block cipher like the Advanced Encryption Standard (AES), we don't need the Initialization Vectors (IVs). Therefore, we just need two Message Authentication Code (MAC) keys for each side that are 16 bytes (128 bits) each since the specified MD5 hash digest size is 16 bytes. In addition, the RC4 cipher uses a 16 byte (128 bit) key that both sides will need as well. All told, we need 2*16 + 2*16 = 64 bytes from the key block.

Running the PRF, we get these values:

> client_write_MAC_secret = 80 B8 F6 09 51 74 EA DB 29 28 EF 6F 9A B8 81 B0
> server_write_MAC_secret = 67 7C 96 7B 70 C5 BC 62 9D 1D 1F 4A A6 79 81 61
> client_write_key = 32 13 2C DD 1B 39 36 40 84 4A DE E5 6C 52 46 72
> server_write_key = 58 36 C4 0D 8C 7C 74 DA 6D B7 34 0A 91 B6 8F A7

**Prepare to be Encrypted!**

The last handshake message the client sends out is the "Finished message." This is a clever message that proves that no one tampered with the handshake and it proves that we know the key. The client takes all bytes from all



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 295
Keysight, EX-1045
Page 295

handshake messages and puts them into a "handshake_messages" buffer. We then calculate 12 bytes of "verify_data" using the pseudo-random function (PRF) with our master key, the label "client finished", and an MD5 and SHA-1 hash of "handshake_messages":

> verify_data = PRF(master_secret, "client finished", MD5(handshake_messages) + SHA-1(handshake_messages)) [12]

We take the result and add a record header byte "0x14" to indicate "finished" and length bytes "00 00 0c" to indicate that we're sending 12 bytes of verify data. Then, like all future encrypted messages, we need to make sure the decrypted contents haven't been tampered with. Since our cipher suite in use is TLS_RSA_WITH_RC4_128_MD5, this means we use the MD5 hash function.

Some people get paranoid when they hear MD5 because it has some weaknesses. I certainly don't advocate using it as-is. However, TLS is smart in that it doesn't use MD5 directly, but rather the HMAC version of it. This means that instead of using MD5(m) directly, we calculate:

> HMAC_MD5(Key, m) = MD5((Key ⊕ opad) ++ MD5((Key ⊕ ipad) ++ m)

(The ⊕ means XOR, ++ means concatenate, "opad" is the bytes "5c 5c ... 5c", and "ipad" is the bytes "36 36 ... 36").

In particular, we calculate:

> HMAC_MD5(client_write_MAC_secret, seq_num + TLSCompressed.type + TLSCompressed.version + TLSCompressed.length + TLSCompressed.fragment));

As you can see, we include a sequence number ("seq_num") along with attributes of the plaintext message (here it's called "TLSCompressed"). The sequence number foils attackers who might try to take a previously encrypted message and insert it midstream. If this occurred, the sequence numbers would definitely be different than what we expected. This also protects us from an attacker dropping a message.

All that's left is to encrypt these bytes.

**RC4 Encryption**

Our negotiated cipher suite was TLS_RSA_WITH_RC4_128_MD5. This tells us that we need to use Ron's Code #4 (RC4) to encrypt the traffic. Ron Rivest developed the RC4 algorithm to generate random bytes based on a 256 byte key. The algorithm is so simple you can actually memorize it in a few minutes.

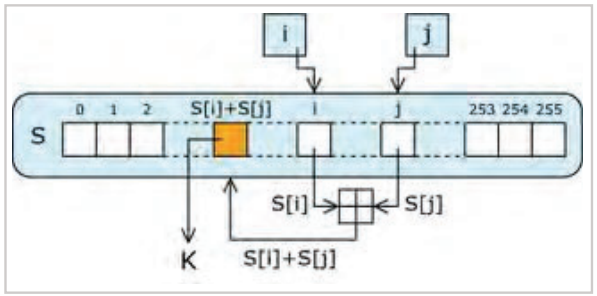
RC4 begins by creating a 256-byte "S" byte array and populating it with 0 to 255. You then iterate over the array by mixing in bytes from the key. You do this to create a state machine that is used to generate "random" bytes. To



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 296
Keysight, EX-1045
Page 296

generate a random byte, we shuffle around the "S" array.

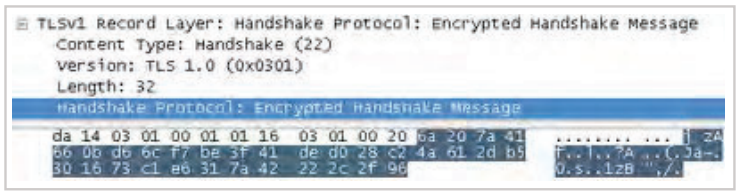
Put graphically, it looks like this:




To encrypt a byte, we xor this pseudo-random byte with the byte we want to encrypt. Remember that xor'ing a bit with 1 causes it to flip. Since we're generating random numbers, on average the xor will flip half of the bits. This random bit flipping is effectively how we encrypt data. As you can see, it's not very complicated and thus it runs quickly. I think that's why Amazon chose it.

Recall that we have a "client_write_key" and a "server_write_key." The means we need to create two RC4 instances: one to encrypt what our browser sends and the other to decrypt what the server sent us.

The first few random bytes out of the "client_write" RC4 instance are "7E 20 7A 4D FE FB 78 A7 33 ..." If we xor these bytes with the unencrypted header and verify message bytes of "14 00 00 0C 98 F0 AE CB C4 ...", we'll get what appears in the encrypted portion that we can see in Wireshark:




The server does almost the same thing. It sends out a "Change Cipher Spec" and then a "Finished Message" that includes all handshake messages, including the *decrypted* version of the client's "Finished Message." Consequently, this proves to the client that the server was able to successfully decrypt our message.

**Welcome to the Application Layer!**

Now, 220 milliseconds after we started, we're finally ready for the application layer. We can now send normal HTTP traffic that'll be encrypted by the TLS layer with the RC4 write instance and decrypt traffic with the server RC4 write instance. In addition, the TLS layer will check each record for tampering by computing the HMAC_MD5 hash of the contents.

At this point, the handshake is over. Our TLS record's content type is now 23 (0x17). Encrypted traffic begins with "17 03 01" which indicate the record type



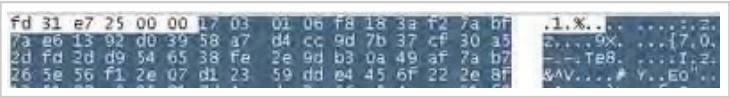
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 297
Keysight, EX-1045
Page 297

and TLS version. These bytes are followed by our encrypted size, which includes the HMAC hash.

Encrypting the plaintext of:

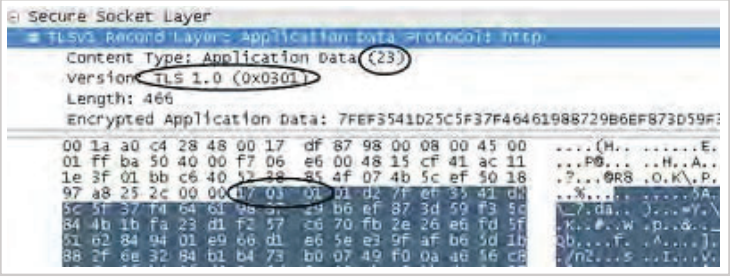
> GET /gp/cart/view.html/ref=pd_luc_mri HTTP/1.1
> Host: www.amazon.com
> User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.9.0.10) Gecko/2009060911 Minefield/3.0.10 (.NET CLR 3.5.30729)
> Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
> Accept-Language: en-us,en;q=0.5
> Accept-Encoding: gzip,deflate
> Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
> Keep-Alive: 300
> Connection: keep-alive
> ...

will give us the bytes we see on the wire:




The only other interesting fact is that the sequence number increases on each record, it's now 1 (and the next record will be 2, etc).

The server does the same type of thing on its side using the server_write_key. We see its response, including the tell-tale application data header:




Decrypting this gives us:

> HTTP/1.1 200 OK
> Date: Wed, 10 Jun 2009 01:09:30 GMT
> Server: Server
> ...
> Cneonction: close
> Transfer-Encoding: chunked

which is a normal HTTP reply that includes a non-descriptive "Server: Server"



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 298
Keysight, EX-1045
Page 298

header and a misspelled "Cneonction: close" header coming from Amazon's load balancers.

TLS is just below the application layer. The HTTP server software can act as if it's sending unencrypted traffic. The only change is that it writes to a library that does all the encryption. OpenSSL is a popular open-source library for TLS.

The connection will stay open while both sides send and receive encrypted data until either side sends out a "closure alert" message and then closes the connection. If we reconnect shortly after disconnecting, we can re-use the negotiated keys (if the server still has them cached) without using public key operations, otherwise we do a completely new full handshake.

It's important to realize that application data records can be *anything*. The only reason "HTTPS" is special is because the web is so popular. There are lots of other TCP/IP based protocols that ride on top of TLS. For example, TLS is used by FTPS and secure extensions to SMTP. It's certainly better to use TLS than inventing your own solution. Additionally, you'll benefit from a protocol that has withstood careful security analysis.

**... And We're Done!**

The very readable TLS RFC covers many more details that were missed here. We covered just one single path in our observation of the 220 millisecond dance between Firefox and Amazon's server. Quite a bit of the process was affected by the TLS_RSA_WITH_RC4_128_MD5 Cipher Suite selection that Amazon made with its ServerHello message. It's a reasonable choice that slightly favors speed over security.

As we saw, if someone could secretly factor Amazon's "n" modulus into its respective "p" and "q", they could effectively decrypt all "secure" traffic until Amazon changes their certificate. Amazon counter-balances this concern this with a short one year duration certificate:




One of the cipher suites that was offered was "TLS_DHE_RSA_WITH_AES_256_CBC_SHA" which uses the Diffie-Hellman key exchange that has a nice property of "forward secrecy." This means that if someone cracked the mathematics of the key exchange, they'd be no better off to decrypt another session. One downside to this algorithm is that it requires more math with big numbers, and thus is a little more computationally taxing on a busy server. The "Advanced Encryption Standard" (AES) algorithm was present in many of the suites that we offered. It's different than RC4 in that it works on 16 byte "blocks" at a time rather



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 299
Keysight, EX-1045
Page 299

than a single byte. Since its key can be up to 256 bits, many consider this to be more secure than RC4.

In just 220 milliseconds, two endpoints on the Internet came together, provided enough credentials to trust each other, set up encryption algorithms, and started to send encrypted traffic.

And to think, all of this just so Bob can buy milk.

POSTED BY JEFF MOSER AT 8:57 AM

### 80 COMMENTS:

Drew said...

Great post, Jeff. I'm curious to know if there is a way to disable certain algorithms in FF? Could you cut down that list of 34 algorithms? What will Amazon choose if you make FF refuse to use TLS_RSA_WITH_RC4_128_MD5?

JUNE 10, 2009 10:01 AM

Jeff Moser said...

Drew: You can disable cipher suites by typing "about:config" in your address bar and then filter for "RSA"

You'll see a huge list, including "security.ssl3.rsa_rc4_128_md5", if you set this to false and then go back to Amazon.com, you'll see that it now picks "TLS_RSA_WITH_AES_256_CBC_SHA"

Amazon is willing to spend extra time using 256 bit AES, but they still really don't like the expensive Diffie-Hellman key exchange :)

JUNE 10, 2009 10:06 AM

Eddie P said...

Thanks.. now i feel as dumb as a rock... excuse me while i go stare blankly at the walll

JUNE 10, 2009 10:18 AM

Will Shaver said...

This is a fantastic post. Thanks for taking the time to write it!

JUNE 10, 2009 10:26 AM

robburke said...

Really superb post -- tremendously informative and an engaging read!! Thank



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 300
Keysight, EX-1045
Page 300

you Jeff.

JUNE 10, 2009 10:56 AM

Anonymous said...

I learned something, thank you!

JUNE 10, 2009 11:11 AM

Anonymous said...

Thanks for this post, Jeff. Great stuff.

JUNE 10, 2009 11:53 AM

Jim C said...

I love it when somebody like yourself takes the time to make an ambitious post like this one - thanks for contributing this body of information to the searchable web.

- you've likely helped more people than you think you have...

JUNE 10, 2009 12:00 PM

sriram srinivasan said...

Thanks much for the lovely presentation. This is great work.

JUNE 10, 2009 12:23 PM

Eric Frenkiel said...

Fantastic post, Jeff! Your ability to clearly explain difficult concepts is unmatched =) I look forward to dropping by more often.

JUNE 10, 2009 12:28 PM

Anonymous said...

Thanks very much for taking the time to post this article.. Truly fascinating and informative read.

JUNE 10, 2009 12:29 PM

Mr T's Fashion Consultant said...

Great article, very informative. I like Wireshark too. Can you tell us how you launched Wireshark (command line options or whatever) to get the captured data? Thanks for any info.

JUNE 10, 2009 12:39 PM

Adam said...



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 301
Keysight, EX-1045
Page 301

At least with newer versions of FireFox, you left out an important part of many new ecommerce HTTPS connections - OCSP verification of the validity of the certificate from the CA itself, in real time.
The particular Amazon cert you used didn't make use of this (only the new EV certs do), but for the 'fully secure' new status bar and color change in FF3+ and IE8, you have to have an EV cert with valid OCSP responder.

JUNE 10, 2009 12:41 PM

Gabe Sumner said...

> If nothing else, this extension should allow an extra week or so of IPv4 addresses.

:) Made me chuckle!

JUNE 10, 2009 12:44 PM

chkno said...

Note: RFC 4217 FTP with TLS is more often referred to as FTPS rather than SFTP. SFTP is more commonly used to refer to the SSH File Transfer Protocol.

JUNE 10, 2009 12:57 PM

John Doe said...

DH is not preferred because its costlier, it only does the key-exchange so authentication still needs to be done using DSA or RSA (hence called TLS_DHE_RSA_WITH_AES_256_CBC_SHA), while RSA can be used for both key-exchange and auth. DH additionally might require 'server-key-exchange' SSL messages. Lastly, most crypto-cards support a full RSA handshake operation which can't be used for DH.

JUNE 10, 2009 12:58 PM

C.Santini said...

Thank you! Great post!!
my little contribute for RSA understanding:
RSA encryption by hand :)

JUNE 10, 2009 1:22 PM

Anonymous said...

Thank you.

JUNE 10, 2009 1:30 PM

Tony Oppenheim said...



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 302
Keysight, EX-1045
Page 302

Wonderful post! Coincidently I've just been reading Steven Levy's great book "Crypto: How the Code Rebels Beat the Government Saving Privacy in the Digital Age" which tells the backstory of how RSA and many of the other encryption methods we now use came to be.

Thanks for posting!

JUNE 10, 2009 1:39 PM

Anonymous said...

I read about week ago about SRP protocol. It is so brillant. It would be really nice, if it will be supported in browsers and webservers.

Nice post.

JUNE 10, 2009 2:24 PM

Anonymous said...

Excellent write up. Thank you for putting this together!!

JUNE 10, 2009 2:25 PM

Anonymous said...

What an excellent walkthrough. I was geekily exhilarating. You took the time to annotate this beautifully so the many might enjoy it. Wunderbar.

JUNE 10, 2009 3:37 PM

Anonymous said...

always mind your p's and q's

JUNE 10, 2009 3:42 PM

JeremyH said...

Nice post. One other thing that goes on even before the HTTPS connect is a DNS lookup to connect to a server called www.amazon.com. Mozilla will first see if 'www.amazon.com' is in the DNS cache and if not will start to recursively resolve the domain name, starting at the root ('.'), then the TLD ('.com.'), then the domain ('amazon.com.'). It all happens before the initial TCP connection. This looks like this:

; <<>> DiG 9.4.2 <<>> +trace www.amazon.com
;; global options: printcmd
. 297443 IN NS J.ROOT-SERVERS.NET.
. 297443 IN NS L.ROOT-SERVERS.NET.
. 297443 IN NS D.ROOT-SERVERS.NET.
. 297443 IN NS G.ROOT-SERVERS.NET.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 303
Keysight, EX-1045
Page 303

. 297443 IN NS I.ROOT-SERVERS.NET.
. 297443 IN NS K.ROOT-SERVERS.NET.
. 297443 IN NS M.ROOT-SERVERS.NET.
. 297443 IN NS B.ROOT-SERVERS.NET.
. 297443 IN NS F.ROOT-SERVERS.NET.
. 297443 IN NS H.ROOT-SERVERS.NET.
. 297443 IN NS C.ROOT-SERVERS.NET.
. 297443 IN NS A.ROOT-SERVERS.NET.
. 297443 IN NS E.ROOT-SERVERS.NET.
;; Received 512 bytes from 68.87.71.226#53(68.87.71.226) in 32 ms

com. 172800 IN NS a.gtld-servers.net.
com. 172800 IN NS b.gtld-servers.net.
com. 172800 IN NS c.gtld-servers.net.
com. 172800 IN NS d.gtld-servers.net.
com. 172800 IN NS e.gtld-servers.net.
com. 172800 IN NS f.gtld-servers.net.
com. 172800 IN NS g.gtld-servers.net.
com. 172800 IN NS h.gtld-servers.net.
com. 172800 IN NS i.gtld-servers.net.
com. 172800 IN NS j.gtld-servers.net.
com. 172800 IN NS k.gtld-servers.net.
com. 172800 IN NS l.gtld-servers.net.
com. 172800 IN NS m.gtld-servers.net.
;; Received 492 bytes from 199.7.83.42#53(L.ROOT-SERVERS.NET) in 60 ms

amazon.com. 172800 IN NS udns1.ultradns.net.
amazon.com. 172800 IN NS udns2.ultradns.net.
;; Received 116 bytes from 192.35.51.30#53(f.gtld-servers.net) in 102 ms

www.amazon.com. 887 IN NS ns-923.amazon.com.
www.amazon.com. 887 IN NS ns-921.amazon.com.
www.amazon.com. 887 IN NS ns-912.amazon.com.
www.amazon.com. 887 IN NS ns-911.amazon.com.
;; Received 180 bytes from 204.74.101.1#53(udns2.ultradns.net) in 20 ms

www.amazon.com. 60 IN A 72.21.207.65
;; Received 48 bytes from 72.21.192.209#53(ns-921.amazon.com) in 23 ms

JUNE 10, 2009 4:20 PM

spdalton said...

Brilliant post! Was a great way to see real application for the introduction to networking course I'm taking. Thanks!



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 304
Keysight, EX-1045
Page 304

JUNE 10, 2009 4:54 PM

Anonymous said...

Great post! Learnt something new today. Thanks.

JUNE 10, 2009 5:31 PM

Anonymous said...

Great post!

JUNE 10, 2009 6:32 PM

Anonymous said...

FTP over SSL is called FTPS. There's enough confusion already regarding SFTP vs. FTPS so please fix your mistake.

JUNE 10, 2009 7:10 PM

Vicente said...

Nice article!

JUNE 10, 2009 9:57 PM

Jan Michael Yu said...

Awesome post. My mind went blank halfway through but I'm saving this for a later read. I wish classes were taught like this in school.

JUNE 10, 2009 10:31 PM

Jeff Moser said...

Will Shaver, robburke, Anonymous #1/#2/#3/#4/#5/#7/#10/#11, sriaram srinivasan, and Vicente: Thanks for the postive feedback! I really appreciate all of them. As I was writing this post over the past four weeks, I was worried that it'd be too long and no one would read it (even after I cut a few thousand words). It's been very encouraging to see the response.

Eddie P: That wasn't the intent -- hopefully TLS is more exciting than staring at a wall :)

Jim C: Thanks for the encouragement. My hope is that this might encourage others to see the beautiful protocols that underly everyday Internet usage.

Eric Frenkiel - It'd be great to see you back. Feel free to click the "Subscribe" buttons on the right hand side (either via RSS or email)

Mr T's Fashion Consultant: I downloaded Wireshark from wireshark.org,



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 305
Keysight, EX-1045
Page 305

installed it, and then ran it. I then clicked "Capture", then "Interfaces" and clicked "Start" on the interface that was generating packets. Once you do this, it'll start to capture all traffic. I then ran the debug Firefox build (Minefield) and went to Amazon. After all this I clicked "Capture", then "Stop". In the protocol column, I found the first "TLSv1" protocol packet and then right-clicked and then clicked on "Follow TCP Stream." Everything else came from looking at the specific packets and seeing the Wireshark decoding of the bytes.

Adam: Very true on OCSP's, CRL's, and the EV extended checks.

Gabe Sumner: Glad you enjoyed it - I tried to keep things lighthearted :-)

chkno/Anonymous #12: You're right. I've gone ahead and made the change from SFTP to FTPS.

John Doe: Point taken on DH for just key exchange and RSA for authentication. This is honestly what I meant as I had linked to the appropriate Wikipedia articles. What I was trying to indicate was DH is nice from a "forward secrecy" perspective. The "downsides" if you can call them that, are more math == more CPU and you have ephemeral keys which means that a network admin has a harder time snooping on the traffic.

C.Santini: Thanks. What do you think of the math with huge numbers? :)

Tony Oppenheim: Sounds like an interesting book. Did anything in particular stick out from the backstory that you thought was interesting?

Anonymous #6: SRP is one of those hidden Internet gems. It's too bad it's not more popular (and therefore, available in more products).

Anonymous #8: Nice use of "geekily exhilirating" :)

Anonymous #9: I don't mind the p's and q's, just so long as they're prime.

JeremyH: Yep, the DNS query answer came in right before the Client Hello. One thing that I didn't cover here is DNS-SEC, which uses certificates to sign responses. It's not widely deployed (yet), but it helps to prevent against DNS cache poisioning.

spdalton: Glad it was helpful. Computers offer the ultimate lab science. I'd highly encourage you to play around with Wireshark to see what's happening on your own network. I think it's fun to see what's actually going on.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 306
**Keysight, EX-1045**
**Page 306**

Jan Michael Yu: Don't worry if it takes a little for it to sink in. It took me a few weeks of exploring to feel comfortable with it. Feel free to ask follow-up questions if something doesn't make sense.

JUNE 10, 2009 11:20 PM

Dom said...

You no doubt have one already but you could probably submit this as a Masters paper and come away with Honors from most top universities. Nice work!

JUNE 11, 2009 12:13 AM

sanjaya said...

wow.. this is amazing though takes time to get into bits and bytes, but now have some idea about how protocol works.

JUNE 11, 2009 12:42 AM

Anonymous said...

Great post! Thanks dude!

JUNE 11, 2009 1:14 AM

pats said...

Everytime i read that secure traffic flows thro' a tunnel, are all these data packets goes through 'tunnel' ? ... my defination of tunnel is something that connects client and server and no one can read actual data flowing throgh since it is encrypted ... if wrong correct me..

JUNE 11, 2009 2:23 AM

Tony Oppenheim said...

@Jeff Moser: "Did anything in particular stick out from the backstory that you thought was interesting?"

Mostly that the NSA did everything they could to keep RSA and other private sector encryption methods from coming into use. It's thru the efforts of really just a few dedicated/brilliant/eccentric/obsessed people that we have the level of security for commercial use we enjoy today. If it wasn't for them e-commerce as we know it wouldn't be happening.

Or at least that's how it seems from my reading. Though I'm no expert in this field.

JUNE 11, 2009 2:49 AM



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 307
Keysight, EX-1045
Page 307

Anonymous said...

This is monumental work, Jeff. I would love to read more from you.

JUNE 11, 2009 2:50 AM

Siu said...

Thank you for the post! Very useful!

I will need to print this to understand everythig.

JUNE 11, 2009 5:07 AM

Anonymous said...

Thanks a lot this enlightening article. It is both detailed and technically correct, as well as understandable. Very well structured and written from a didactical point of view.

JUNE 11, 2009 5:32 AM

Anonymous said...

Awesome article. Please make a pretty PDF version of it!

JUNE 11, 2009 6:04 AM

John Doe said...

@Jeff - Should submit it to phrack. Would be real cool to have a phrack entry!

JUNE 11, 2009 6:10 AM

Niranjan said...

Excellent post, very informative and easy to understand!

JUNE 11, 2009 6:13 AM

Nguyen said...

such a good research :D

JUNE 11, 2009 6:44 AM

earthgecko said...

Fascinating article, thanks. Found the RSA part very interesting. It does just go to demonstrate how much data is flying around... and how quick it does fly around.

JUNE 11, 2009 7:29 AM

Anonymous said...

You are totally awesome for this post, great work!!! Oh, and I almost forgot,



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 308
Keysight, EX-1045
Page 308

Firefox rules! Kudos

JUNE 11, 2009 7:39 AM

Pdizzle said...

Great post thanks a lot. This ties in well with the SSCP class I just took.

JUNE 11, 2009 7:40 AM

Dave said...

I read you article while wearing my Three Wolf Moon T-Shirt (link below), and I was actually able to understand the math ! It's amazing, because the last math class I had was in my senior year of elementary school. Thanks to your wonderful explanation of the SSL handshake, I can now encrypt and decrypt human conversation in real time. I will try using what I have learned here to decrypt the mathematical language of the wolves in order to discover their plan to reach the moon again before the Russians can beat us there. Thank you !

Three Wolf Moon T-Shirt

JUNE 11, 2009 8:16 AM

Anonymous said...

Interesting hack. What are you going to do with this knowledge? Hack Firefox users who access Paypal?

JUNE 11, 2009 8:20 AM

Jeremy said...

Great post, thank you very much. I'm a graduate student in computer science, but my focus is databases so I don't often get to see the networking end of the wire.

JUNE 11, 2009 8:28 AM

Jeff Moser said...

Sanjaya, Anonymous #12/#13/#14/#15, Siu, John Doe, Niranjan, Nguyen, earthgecko, Pdizzle, Jeremy: Thanks for the kind words and spending the time to comment. I appreciate it.

Dom: That'd be nice, but I'm sure it's a bit tougher than that :) No masters for me, just a Bachelor's in Computer Science and Math. I'm just a guy that likes exploring with Wireshark.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 309
Keysight, EX-1045
Page 309

Pats: Tunnels allow normally unencrypted traffic to appear as encrypted when on the wire. SSH is a popular example. You can sort of think of TLS/SSL as a secure "tunnel" for HTTPS traffic.

Tony Oppenheim: Thanks for sharing that. I read on Wikipedia that Clifford Cocks invented a similar system to RSA 4-5 years earlier for the UK intelligence agency GCHQ, but didn't really implement or deploy it. I always enjoy the history behind the technology.

Anonymous #15: Any advice on what you'd like a "prettier PDF version" to look like? Like a magazine (two-columns?)

Dave: glad the shirt gave you its claimed super powers :) It'd be nice if practical math like RSA was taught in high school. I think it'd be more engaging... at least a few weeks of it.

Anonymous #16: I have no hacking intents... it's more of just seeing all these protocols that are used everyday as having an elegance to them.

JUNE 11, 2009 8:46 AM

Anonymous said...

Wow,
this explanation ROCKS!

Can you please do the same for SIP/RTP kind of traffic (in combination with wireshark)?

JUNE 11, 2009 8:56 AM

Sasi said...

wow !! Clear & Well explained ... Thanks Jeff.

JUNE 11, 2009 9:09 AM

Anonymous said...

Great post, thanks! BTW this post has made it to the front page of digg.com

JUNE 11, 2009 9:45 AM

Waan said...

Great post,Jeff.... Thank you!

JUNE 11, 2009 10:19 AM

Anonymous said...

I think i learned more from this than my Cisco Security class... and we spent



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 310
Keysight, EX-1045
Page 310

like 3 weeks on this...

JUNE 11, 2009 10:25 AM

JaKe said...

A PDF would be great! Just a simple straight from the page print of the text, graphics and hyperlinks. Nothing fancy, like two-column, required, the graphics won't let you anyway ;) .

JUNE 11, 2009 10:50 AM

CVillalobos said...

Okay. That was cool thanks for the info. Detailed and a fun way to start the morning.

JUNE 11, 2009 11:00 AM

Anonymous said...

Great Info, great article. Thank you so much..

JUNE 11, 2009 11:25 AM

Anonymous said...

Great Article, very intresting and useful Jeff!!!

JUNE 11, 2009 12:02 PM

Ankur said...

Good stuff, Jeff! Truly informative.

JUNE 11, 2009 1:31 PM

Television Online said...

Possibly the most intense milliseconds that you'll never actually notice.

JUNE 11, 2009 3:05 PM

Anonymous said...

wow, you have no life

JUNE 11, 2009 3:41 PM

k@beza said...

Oh my god !! Oh my god !! You can see the Matrix !!
Fuc*ing insaneeeeeeeeeee !

JUNE 11, 2009 3:43 PM

sulumits retsambew said...



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 311
Keysight, EX-1045
Page 311

very deep explanation , thank you

JUNE 11, 2009 4:16 PM

Peter Vatistas said...

Awesome post! I'm a big fan of Wireshark.

JUNE 11, 2009 4:27 PM

David Thomson said...

I read one comment about SRP. I implemented this in a web browser called SupraBrowser

http://sourceforge.net/projects/suprabrowser

It supports encryption of all traffic using 3DES (with pluggable algorithms), based on a Diffie-Hellman key exchange augmented by the result of a bi-directional SRP authentication (http://srp.stanford.edu).

JUNE 11, 2009 4:27 PM

tarmo said...

Good stuff. Well explained.

JUNE 11, 2009 4:38 PM

Anonymous said...

Thanks a lot for the post, it was very informative.

JUNE 11, 2009 6:15 PM

Rajat Swarup said...

Some additional information:
For really large primes p,q such that n=pxq the values of the public exponent e and private exponent d are computed as :
e x d = 1 mod (p-1)x(q-1)
The value (p-1) x (q-1) is also called as the Euler's totient function. :-)

JUNE 11, 2009 6:16 PM

Rajat Swarup said...

Btw...I forgot to mention...this was an awesome post! :-)

JUNE 11, 2009 6:17 PM

Anonymous said...

Thanks for a thorough and ... just great post. Awesome!



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 312
Keysight, EX-1045
Page 312

JUNE 11, 2009 6:24 PM

Anonymous said...

@Mr.Moser... So for example, the 'client_write_mac_secret' is an associative array, whose hash_value's value is part or all of the key_block? Am I understanding that part correctly?

By the way, great post! And I don't mean great in the way in which it is normally cheaply used in our day to day activities. Unless blogger.com goes down, I think this post will be indexed highly on the internets after you and I are long gone. You might want to back this info up in a couple other places chief!

JUNE 12, 2009 12:21 AM

Anonymous said...

Outstanding article. In the STARTTLS E-Mail world it is neccessary to force Wireshark to interpret the SMTP stream as SSL, as they don't seem to have found a robust way to detect the transition from plain to encrypted yet. Once this is done, everything works just the same as described here.

JUNE 12, 2009 12:25 AM

Anonymous said...

@Mr.Moser... just for clarification on a typo in my last post a few moments ago, when I said "hash_value"'s value ...i meant "hash_size"'s value. Please clarify if you have a minute. If you explain this one, I will understand the others. It's an associative array, correct? But how is that value populated or determined. I'm foggy on that.

JUNE 12, 2009 12:28 AM

Anonymous said...

wow,
this article is perhaps a reply to a loosely based question 'dad, where did I come from?'

and your explanation is a very good elaboration of facts and figures.

great effort and good job.
well done and yes I am enlightened.

JUNE 12, 2009 2:38 AM

Rajasekhar said...

I think this is one of most clearly written blog-articles I have read in recent



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 313
Keysight, EX-1045
Page 313

times. Thanks a ton for the tech details. I am yet to get a full grasp of it, but when I do, it will definitely add to my tech armor :) Great work!

JUNE 12, 2009 3:27 AM

Suebtas said...

Awesome tutorial SSL article.

JUNE 12, 2009 3:29 AM

Mary-Frances Beesorchard said...

Thank You! This article makes such a clear study-plan for beginners.

JUNE 12, 2009 7:23 AM

Jeff Moser said...

Anonymous #19/#20/#21/#22/#24/#25/#29, Sasi, Waan, CVillalobos, Ankur, sulumits retsambew, Peter Vatistas, tarmo, Rajasekhar, Suebtas, Mary-Frances Beesorchard: Thanks for the kind comments and for stopping by.

Anonymous #18: SIP/RTP is unencrypted and Wireshark handles it fairly well on its own. I'd encourage you to download Wireshark and just take a few seconds of captures and then explore it in Wireshark. That'd probably be more insightful.

JaKe: Per your request, I created a PDF version of this post. Does this work for you?

Television Online: Yeah, I typically don't notice them most times, but now I respect those milliseconds more.

David Thomson: Glad to see use of SRP.

Rajat Swarup: Yep. If you check out the links I had in the RSA section, you'll see that I linked to the totient function along with Fermat's Little Theorem, the Chinese Remainder theorem, and the original RSA paper that outlines this as well.

Anonymous #26/#28: The "client_write_MAC_secret" are the first "hash_size" bytes (in this case, 16 bytes since it's MD5) of "key_block". The "key_block" is the output of the Pseudo Random Function (PRF) that I mentioned. You don't need to hash it any more. This is the "key" of the keyed-HMAC version of MD5 that I described in the post.

For completeness:

client_write_MAC_secret are bytes 0..[SecurityParameters.hash_size] - 1 of the PRF generated "key_block"
server_write_MAC_secret are bytes [SecurityParameters.hash_size] .. [SecurityParameters.hash_size + SecurityParameters.hash_size] - 1 of the PRF generated "key_block"
client_write_key are bytes [SecurityParameters.hash_size + SecurityParameters.hash_size]..[SecurityParameters.hash_size + SecurityParameters.hash_size + SecurityParameters.key_material_length] -1 of the PRF generated "key_block"
server_write_key are bytes [SecurityParameters.hash_size + SecurityParameters.hash_size + SecurityParameters.key_material_length].. [SecurityParameters.hash_size + SecurityParameters.hash_size + SecurityParameters.key_material_length + SecurityParameters.key_material_length] - 1 of the PRF generated "key_block"

Does that help? Feel free to check out the RFC I linked to if you want even more details.

Anonymous #27: It's a little trickier to detect a transition. A good heuristic would be to look for the the TLS version bytes and content types... along with seemingly random data.

JUNE 12, 2009 8:32 AM

Jimmy In Madrid (J.I.M) said...

Amazing! great work there!

JUNE 12, 2009 9:24 AM

Post a Comment

Home Older Post

Subscribe to: Post Comments (Atom)



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 315
Keysight, EX-1045
Page 315

Network Working Group M. Allman
Request for Comments: 2577 NASA Glenn/Sterling Software
Category: Informational S. Ostermann
Ohio University
May 1999

# FTP Security Considerations

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

Abstract

The specification for the File Transfer Protocol (FTP) contains a number of mechanisms that can be used to compromise network security. The FTP specification allows a client to instruct a server to transfer files to a third machine. This third-party mechanism, known as proxy FTP, causes a well known security problem. The FTP specification also allows an unlimited number of attempts at entering a user's password. This allows brute force "password guessing" attacks. This document provides suggestions for system administrators and those implementing FTP servers that will decrease the security problems associated with FTP.

## 1 Introduction

The File Transfer Protocol specification (FTP) [PR85] provides a mechanism that allows a client to establish an FTP control connection and transfer a file between two FTP servers. This "proxy FTP" mechanism can be used to decrease the amount of traffic on the network; the client instructs one server to transfer a file to another server, rather than transferring the file from the first server to the client and then from the client to the second server. This is particularly useful when the client connects to the network using a slow link (e.g., a modem). While useful, proxy FTP provides a security problem known as a "bounce attack" [CERT97:27]. In addition to the bounce attack, FTP servers can be used by attackers to guess passwords using brute force.



https://web.archive.org/web/20070318022322/https://tools.ietf.org/html/rfc2577

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 316
Keysight, EX-1045
Page 316

This document does not contain a discussion of FTP when used in conjunction with strong security protocols, such as IP Security. These security concerns should be documented, however they are out of the scope of this document.

This paper provides information for FTP server implementers and system administrators, as follows. Section 2 describes the FTP "bounce attack". Section 3 provides suggestions for minimizing the bounce attack. Section 4 provides suggestions for servers which limit access based on network address. Section 5 provides recommendations for limiting brute force "password guessing" by clients. Next, section 6 provides a brief discussion of mechanisms to improve privacy. Section 7 provides a mechanism to prevent user identity guessing. Section 8 discusses the practice of port stealing. Finally, section 9 provides an overview of other FTP security issues related to software bugs rather than protocol issues.

## 2 The Bounce Attack

The version of FTP specified in the standard [PR85] provides a method for attacking well known network servers, while making the perpetrators difficult to track down. The attack involves sending an FTP "PORT" command to an FTP server containing the network address and the port number of the machine and service being attacked. At this point, the original client can instruct the FTP server to send a file to the service being attacked. Such a file would contain commands relevant to the service being attacked (SMTP, NNTP, etc.). Instructing a third party to connect to the service, rather than connecting directly, makes tracking down the perpetrator difficult and can circumvent network-address-based access restrictions.

As an example, a client uploads a file containing SMTP commands to an FTP server. Then, using an appropriate PORT command, the client instructs the server to open a connection to a third machine's SMTP port. Finally, the client instructs the server to transfer the uploaded file containing SMTP commands to the third machine. This may allow the client to forge mail on the third machine without making a direct connection. This makes it difficult to track attackers.

## 3 Protecting Against the Bounce Attack

The original FTP specification [PR85] assumes that data connections will be made using the Transmission Control Protocol (TCP) [Pos81]. TCP port numbers in the range 0 - 1023 are reserved for well known services such as mail, network news and FTP control connections [RP94]. The FTP specification makes no restrictions on the TCP port number used for the data connection. Therefore, using proxy FTP,



https://web.archive.org/web/20070318022322/https://tools.ietf.org/html/rfc2577

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 317
Keysight, EX-1045
Page 317

clients have the ability to tell the server to attack a well known service on any machine.

To avoid such bounce attacks, it is suggested that servers not open data connections to TCP ports less than 1024. If a server receives a PORT command containing a TCP port number less than 1024, the suggested response is 504 (defined as "Command not implemented for that parameter" by [PR85]). Note that this still leaves non-well known servers (those running on ports greater than 1023) vulnerable to bounce attacks.

Several proposals (e.g., [AOM98] and [Pis94]) provide a mechanism that would allow data connections to be made using a transport protocol other than TCP. Similar precautions should be taken to protect well known services when using these protocols.

Also note that the bounce attack generally requires that a perpetrator be able to upload a file to an FTP server and later download it to the service being attacked. Using proper file protections will prevent this behavior. However, attackers can also attack services by sending random data from a remote FTP server which may cause problems for some services.

Disabling the PORT command is also an option for protecting against the bounce attack. Most file transfers can be made using only the PASV command [Bel94]. The disadvantage of disabling the PORT command is that one loses the ability to use proxy FTP, but proxy FTP may not be necessary in a particular environment.

## 4 Restricted Access

For some FTP servers, it is desirable to restrict access based on network address. For example, a server might want to restrict access to certain files from certain places (e.g., a certain file should not be transferred out of an organization). In such a situation, the server should confirm that the network address of the remote hosts on both the control connection and the data connection are within the organization before sending a restricted file. By checking both connections, a server is protected against the case when the control connection is established with a trusted host and the data connection is not. Likewise, the client should verify the IP address of the remote host after accepting a connection on a port opened in listen mode to verify that the connection was made by the expected server.

Note that restricting access based on network address leaves the FTP server vulnerable to "spoof" attacks. In a spoof attack, for example, an attacking machine could assume the host address of another machine inside an organization and download files that are



https://web.archive.org/web/20070318022322/https://tools.ietf.org/html/rfc2577

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 318
Keysight, EX-1045
Page 318

not accessible from outside the organization. Whenever possible, secure authentication mechanisms should be used, such as those outlined in [HL97].

## 5 Protecting Passwords

To minimize the risk of brute force password guessing through the FTP server, it is suggested that servers limit the number of attempts that can be made at sending a correct password. After a small number of attempts (3-5), the server should close the control connection with the client. Before closing the control connection the server must send a return code of 421 ("Service not available, closing control connection." [PR85]) to the client. In addition, it is suggested that the server impose a 5 second delay before replying to an invalid "PASS" command to diminish the efficiency of a brute force attack. If available, mechanisms already provided by the target operating system should be used to implement the above suggestions.

An intruder can subvert the above mechanisms by establishing multiple, parallel control connections to a server. To combat the use of multiple concurrent connections, the server could either limit the total number of control connections possible or attempt to detect suspicious activity across sessions and refuse further connections from the site. However, both of these mechanisms open the door to "denial of service" attacks, in which an attacker purposely initiates the attack to disable access by a valid user.

Standard FTP [PR85] sends passwords in clear text using the "PASS" command. It is suggested that FTP clients and servers use alternate authentication mechanisms that are not subject to eavesdropping (such as the mechanisms being developed by the IETF Common Authentication Technology Working Group [HL97]).

## 6 Privacy

All data and control information (including passwords) is sent across the network in unencrypted form by standard FTP [PR85]. To guarantee the privacy of the information FTP transmits, a strong encryption scheme should be used whenever possible. One such mechanism is defined in [HL97].

## 7 Protecting Usernames

Standard FTP [PR85] specifies a 530 response to the USER command when the username is rejected. If the username is valid and a password is required FTP returns a 331 response instead. In order to prevent a malicious client from determining valid usernames on a server, it is suggested that a server always return 331 to the USER command and



https://web.archive.org/web/20070318022322/https://tools.ietf.org/html/rfc2577

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 319
Keysight, EX-1045
Page 319

then reject the combination of username and password for an invalid username.

## 8 Port Stealing

Many operating systems assign dynamic port numbers in increasing order. By making a legitimate transfer, an attacker can observe the current port number allocated by the server and "guess" the next one that will be used. The attacker can make a connection to this port, thus denying another legitimate client the ability to make a transfer. Alternatively, the attacker can steal a file meant for a legitimate user. In addition, an attacker can insert a forged file into a data stream thought to come from an authenticated client. This problem can be mitigated by making FTP clients and servers use random local port numbers for data connections, either by requesting random ports from the operating system or using system dependent mechanisms.

## 9 Software-Base Security Problems

The emphasis in this document is on protocol-related security issues. There are a number of documented FTP security-related problems that are due to poor implementation as well. Although the details of these types of problems are beyond the scope of this document, it should be pointed out that the following FTP features has been abused in the past and should be treated with great care by future implementers:

Anonymous FTP

Anonymous FTP refers to the ability of a client to connect to an FTP server with minimal authentication and gain access to public files. Security problems arise when such a user can read all files on the system or can create files. [CERT92:09] [CERT93:06]

Remote Command Execution

An optional FTP extension, "SITE EXEC", allows clients to execute arbitrary commands on the server. This feature should obviously be implemented with great care. There are several documented cases of the FTP "SITE EXEC" command being used to subvert server security [CERT94:08] [CERT95:16]

Debug Code

Several previous security compromises related to FTP can be attributed to software that was installed with debugging features enabled [CERT88:01].



https://web.archive.org/web/20070318022322/https://tools.ietf.org/html/rfc2577

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 320
Keysight, EX-1045
Page 320

This document recommends that implementors of FTP servers with these capabilities review all of the CERT advisories for attacks on these or similar mechanisms before releasing their software.

## 10 Conclusion

Using the above suggestions can decrease the security problems associated with FTP servers without eliminating functionality.

## 11 Security Considerations

Security issues are discussed throughout this memo.

Acknowledgments

We would like to thank Alex Belits, Jim Bound, William Curtin, Robert Elz, Paul Hethmon, Alun Jones and Stephen Tihor for their helpful comments on this paper. Also, we thank the FTPEXT WG members who gave many useful suggestions at the Memphis IETF meeting.

References

[AOM98] Allman, M., Ostermann, S. and C. Metz, "FTP Extensions for IPv6 and NATs", RFC 2428, September 1998.

[Bel94] Bellovin. S., "Firewall-Friendly FTP", RFC 1579, February 1994.

[CERT88:01] CERT Advisory CA-88:01. ftpd Vulnerability. December, 1988 ftp://info.cert.org/pub/cert_advisories/

[CERT92:09] CERT Advisory CA-92:09. AIX Anonymous FTP Vulnerability. April 27, 1992. ftp://info.cert.org/pub/cert_advisories/

[CERT93:06] CERT Advisory CA-93:06. Wuarchive ftpd Vulnerability. September 19,1997 ftp://info.cert.org/pub/cert_advisories/

[CERT94:08] CERT Advisory CA-94:08. ftpd Vulnerabilities. September 23, 1997. ftp://info.cert.org/pub/cert_advisories/

[CERT95:16] CERT Advisory CA-95:16. wu-ftpd Misconfiguration Vulnerability. September 23, 1997 ftp://info.cert.org/pub/cert_advisories/

[CERT97:27] CERT Advisory CA-97.27. FTP Bounce. January 8, 1998. ftp://info.cert.org/pub/cert_advisories/



https://web.archive.org/web/20070318022322/https://tools.ietf.org/html/rfc2577

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 321
Keysight, EX-1045
Page 321

[HL97] Horowitz, M. and S. Lunt, "FTP Security Extensions", RFC 2228, October 1997.

[Pis94] Piscitello, D., "FTP Operation Over Big Address Records (FOOBAR), RFC 1639, June 1994.

[Pos81] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[PR85] Postel, J. and J. Reynolds, "File Transfer Protocol (FTP)", STD 9, RFC 959, October 1985.

[RP94] Reynolds, J. and J. Postel, "Assigned Numbers", STD 2, RFC 1700, October 1994. See also: http://www.iana.org/numbers.html

## Authors' Addresses

Mark Allman
NASA Glenn Research Center/Sterling Software
21000 Brookpark Rd. MS 54-2
Cleveland, OH 44135

EMail: mallman@grc.nasa.gov

Shawn Ostermann
School of Electrical Engineering and Computer Science
Ohio University
416 Morton Hall
Athens, OH 45701

EMail: ostermann@cs.ohiou.edu



https://web.archive.org/web/20070318022322/https://tools.ietf.org/html/rfc2577

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 322
Keysight, EX-1045
Page 322

Full Copyright Statement

Copyright (C) The Internet Society (1999). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.



https://web.archive.org/web/20070318022322/https://tools.ietf.org/html/rfc2577

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 323
Keysight, EX-1045
Page 323

https://web.archive.org/web/20070318022322/https://tools.ietf.org/html/rfc2577

Network Working Group
Request for Comments: 3550
Obsoletes: 1889
Category: Standards Track

H. Schulzrinne
Columbia University
S. Casner
Packet Design
R. Frederick
Blue Coat Systems Inc.
V. Jacobson
Packet Design

July 2003

# RTP: A Transport Protocol for Real-Time Applications

## Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Copyright Notice

Copyright (C) The Internet Society (2003). All Rights Reserved.

## Abstract

This memorandum describes RTP, the real-time transport protocol. RTP provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services. RTP does not address resource reservation and does not guarantee quality-of-service for real-time services. The data transport is augmented by a control protocol (RTCP) to allow monitoring of the data delivery in a manner scalable to large multicast networks, and to provide minimal control and identification functionality. RTP and RTCP are designed to be independent of the underlying transport and network layers. The protocol supports the use of RTP-level translators and mixers.

Most of the text in this memorandum is identical to RFC 1889 which it obsoletes. There are no changes in the packet formats on the wire, only changes to the rules and algorithms governing how the protocol is used. The biggest change is an enhancement to the scalable timer algorithm for calculating when to send RTCP packets in order to minimize transmission in excess of the intended rate when many participants join a session simultaneously.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 325
Keysight, EX-1045
Page 325

Table of Contents


1. Introduction ..... 4
   1.1 Terminology ..... 5
2. RTP Use Scenarios ..... 5
   2.1 Simple Multicast Audio Conference ..... 6
   2.2 Audio and Video Conference ..... 7
   2.3 Mixers and Translators ..... 7
   2.4 Layered Encodings ..... 8
3. Definitions ..... 8
4. Byte Order, Alignment, and Time Format ..... 12
5. RTP Data Transfer Protocol ..... 13
   5.1 RTP Fixed Header Fields ..... 13
   5.2 Multiplexing RTP Sessions ..... 16
   5.3 Profile-Specific Modifications to the RTP Header ..... 18
      5.3.1 RTP Header Extension ..... 18
6. RTP Control Protocol -- RTCP ..... 19
   6.1 RTCP Packet Format ..... 21
   6.2 RTCP Transmission Interval ..... 24
      6.2.1 Maintaining the Number of Session Members ..... 28
   6.3 RTCP Packet Send and Receive Rules ..... 28
      6.3.1 Computing the RTCP Transmission Interval ..... 29
      6.3.2 Initialization ..... 30
      6.3.3 Receiving an RTP or Non-BYE RTCP Packet ..... 31
      6.3.4 Receiving an RTCP BYE Packet ..... 31
      6.3.5 Timing Out an SSRC ..... 32
      6.3.6 Expiration of Transmission Timer ..... 32
      6.3.7 Transmitting a BYE Packet ..... 33
      6.3.8 Updating we_sent ..... 34
      6.3.9 Allocation of Source Description Bandwidth ..... 34
   6.4 Sender and Receiver Reports ..... 35
      6.4.1 SR: Sender Report RTCP Packet ..... 36
      6.4.2 RR: Receiver Report RTCP Packet ..... 42
      6.4.3 Extending the Sender and Receiver Reports ..... 42
      6.4.4 Analyzing Sender and Receiver Reports ..... 43
   6.5 SDES: Source Description RTCP Packet ..... 45
      6.5.1 CNAME: Canonical End-Point Identifier SDES Item . 46
      6.5.2 NAME: User Name SDES Item ..... 48
      6.5.3 EMAIL: Electronic Mail Address SDES Item ..... 48
      6.5.4 PHONE: Phone Number SDES Item ..... 49
      6.5.5 LOC: Geographic User Location SDES Item ..... 49
      6.5.6 TOOL: Application or Tool Name SDES Item ..... 49
      6.5.7 NOTE: Notice/Status SDES Item ..... 50
      6.5.8 PRIV: Private Extensions SDES Item ..... 50
   6.6 BYE: Goodbye RTCP Packet ..... 51
   6.7 APP: Application-Defined RTCP Packet ..... 52
7. RTP Translators and Mixers ..... 53
   7.1 General Description ..... 53




https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 326
Keysight, EX-1045
Page 326

7.2 RTCP Processing in Translators ..... 55
7.3 RTCP Processing in Mixers ..... 57
7.4 Cascaded Mixers ..... 58
8. SSRC Identifier Allocation and Use ..... 59
8.1 Probability of Collision ..... 59
8.2 Collision Resolution and Loop Detection ..... 60
8.3 Use with Layered Encodings ..... 64
9. Security ..... 65
9.1 Confidentiality ..... 65
9.2 Authentication and Message Integrity ..... 67
10. Congestion Control ..... 67
11. RTP over Network and Transport Protocols ..... 68
12. Summary of Protocol Constants ..... 69
12.1 RTCP Packet Types ..... 70
12.2 SDES Types ..... 70
13. RTP Profiles and Payload Format Specifications ..... 71
14. Security Considerations ..... 73
15. IANA Considerations ..... 73
16. Intellectual Property Rights Statement ..... 74
17. Acknowledgments ..... 74
Appendix A. Algorithms ..... 75
Appendix A.1 RTP Data Header Validity Checks ..... 78
Appendix A.2 RTCP Header Validity Checks ..... 82
Appendix A.3 Determining Number of Packets Expected and Lost ... 83
Appendix A.4 Generating RTCP SDES Packets ..... 84
Appendix A.5 Parsing RTCP SDES Packets ..... 85
Appendix A.6 Generating a Random 32-bit Identifier ..... 85
Appendix A.7 Computing the RTCP Transmission Interval ..... 87
Appendix A.8 Estimating the Interarrival Jitter ..... 94
Appendix B. Changes from RFC 1889 ..... 95
References ..... 100
Normative References ..... 100
Informative References ..... 100
Authors' Addresses ..... 103
Full Copyright Statement ..... 104



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 327
Keysight, EX-1045
Page 327

## 1. Introduction

This memorandum specifies the real-time transport protocol (RTP), which provides end-to-end delivery services for data with real-time characteristics, such as interactive audio and video. Those services include payload type identification, sequence numbering, timestamping and delivery monitoring. Applications typically run RTP on top of UDP to make use of its multiplexing and checksum services; both protocols contribute parts of the transport protocol functionality. However, RTP may be used with other suitable underlying network or transport protocols (see Section 11). RTP supports data transfer to multiple destinations using multicast distribution if provided by the underlying network.

Note that RTP itself does not provide any mechanism to ensure timely delivery or provide other quality-of-service guarantees, but relies on lower-layer services to do so. It does not guarantee delivery or prevent out-of-order delivery, nor does it assume that the underlying network is reliable and delivers packets in sequence. The sequence numbers included in RTP allow the receiver to reconstruct the sender's packet sequence, but sequence numbers might also be used to determine the proper location of a packet, for example in video decoding, without necessarily decoding packets in sequence.

While RTP is primarily designed to satisfy the needs of multi-participant multimedia conferences, it is not limited to that particular application. Storage of continuous data, interactive distributed simulation, active badge, and control and measurement applications may also find RTP applicable.

This document defines RTP, consisting of two closely-linked parts:

- the real-time transport protocol (RTP), to carry data that has real-time properties.

- the RTP control protocol (RTCP), to monitor the quality of service and to convey information about the participants in an on-going session. The latter aspect of RTCP may be sufficient for "loosely controlled" sessions, i.e., where there is no explicit membership control and set-up, but it is not necessarily intended to support all of an application's control communication requirements. This functionality may be fully or partially subsumed by a separate session control protocol, which is beyond the scope of this document.

RTP represents a new style of protocol following the principles of application level framing and integrated layer processing proposed by Clark and Tennenhouse [10]. That is, RTP is intended to be malleable



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 328
Keysight, EX-1045
Page 328

to provide the information required by a particular application and will often be integrated into the application processing rather than being implemented as a separate layer. RTP is a protocol framework that is deliberately not complete. This document specifies those functions expected to be common across all the applications for which RTP would be appropriate. Unlike conventional protocols in which additional functions might be accommodated by making the protocol more general or by adding an option mechanism that would require parsing, RTP is intended to be tailored through modifications and/or additions to the headers as needed. Examples are given in Sections 5.3 and 6.4.3.

Therefore, in addition to this document, a complete specification of RTP for a particular application will require one or more companion documents (see Section 13):

- a profile specification document, which defines a set of payload type codes and their mapping to payload formats (e.g., media encodings). A profile may also define extensions or modifications to RTP that are specific to a particular class of applications. Typically an application will operate under only one profile. A profile for audio and video data may be found in the companion RFC 3551 [1].

- payload format specification documents, which define how a particular payload, such as an audio or video encoding, is to be carried in RTP.

A discussion of real-time services and algorithms for their implementation as well as background discussion on some of the RTP design decisions can be found in [11].

### 1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119 [2] and indicate requirement levels for compliant RTP implementations.

## 2. RTP Use Scenarios

The following sections describe some aspects of the use of RTP. The examples were chosen to illustrate the basic operation of applications using RTP, not to limit what RTP may be used for. In these examples, RTP is carried on top of IP and UDP, and follows the conventions established by the profile for audio and video specified in the companion RFC 3551.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 329
Keysight, EX-1045
Page 329

### 2.1 Simple Multicast Audio Conference

A working group of the IETF meets to discuss the latest protocol document, using the IP multicast services of the Internet for voice communications. Through some allocation mechanism the working group chair obtains a multicast group address and pair of ports. One port is used for audio data, and the other is used for control (RTCP) packets. This address and port information is distributed to the intended participants. If privacy is desired, the data and control packets may be encrypted as specified in Section 9.1, in which case an encryption key must also be generated and distributed. The exact details of these allocation and distribution mechanisms are beyond the scope of RTP.

The audio conferencing application used by each conference participant sends audio data in small chunks of, say, 20 ms duration. Each chunk of audio data is preceded by an RTP header; RTP header and data are in turn contained in a UDP packet. The RTP header indicates what type of audio encoding (such as PCM, ADPCM or LPC) is contained in each packet so that senders can change the encoding during a conference, for example, to accommodate a new participant that is connected through a low-bandwidth link or react to indications of network congestion.

The Internet, like other packet networks, occasionally loses and reorders packets and delays them by variable amounts of time. To cope with these impairments, the RTP header contains timing information and a sequence number that allow the receivers to reconstruct the timing produced by the source, so that in this example, chunks of audio are contiguously played out the speaker every 20 ms. This timing reconstruction is performed separately for each source of RTP packets in the conference. The sequence number can also be used by the receiver to estimate how many packets are being lost.

Since members of the working group join and leave during the conference, it is useful to know who is participating at any moment and how well they are receiving the audio data. For that purpose, each instance of the audio application in the conference periodically multicasts a reception report plus the name of its user on the RTCP (control) port. The reception report indicates how well the current speaker is being received and may be used to control adaptive encodings. In addition to the user name, other identifying information may also be included subject to control bandwidth limits. A site sends the RTCP BYE packet (Section 6.6) when it leaves the conference.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 330
Keysight, EX-1045
Page 330

### 2.2 Audio and Video Conference

If both audio and video media are used in a conference, they are transmitted as separate RTP sessions. That is, separate RTP and RTCP packets are transmitted for each medium using two different UDP port pairs and/or multicast addresses. There is no direct coupling at the RTP level between the audio and video sessions, except that a user participating in both sessions should use the same distinguished (canonical) name in the RTCP packets for both so that the sessions can be associated.

One motivation for this separation is to allow some participants in the conference to receive only one medium if they choose. Further explanation is given in Section 5.2. Despite the separation, synchronized playback of a source's audio and video can be achieved using timing information carried in the RTCP packets for both sessions.

### 2.3 Mixers and Translators

So far, we have assumed that all sites want to receive media data in the same format. However, this may not always be appropriate. Consider the case where participants in one area are connected through a low-speed link to the majority of the conference participants who enjoy high-speed network access. Instead of forcing everyone to use a lower-bandwidth, reduced-quality audio encoding, an RTP-level relay called a mixer may be placed near the low-bandwidth area. This mixer resynchronizes incoming audio packets to reconstruct the constant 20 ms spacing generated by the sender, mixes these reconstructed audio streams into a single stream, translates the audio encoding to a lower-bandwidth one and forwards the lower-bandwidth packet stream across the low-speed link. These packets might be unicast to a single recipient or multicast on a different address to multiple recipients. The RTP header includes a means for mixers to identify the sources that contributed to a mixed packet so that correct talker indication can be provided at the receivers.

Some of the intended participants in the audio conference may be connected with high bandwidth links but might not be directly reachable via IP multicast. For example, they might be behind an application-level firewall that will not let any IP packets pass. For these sites, mixing may not be necessary, in which case another type of RTP-level relay called a translator may be used. Two translators are installed, one on either side of the firewall, with the outside one funneling all multicast packets received through a secure connection to the translator inside the firewall. The translator inside the firewall sends them again as multicast packets to a multicast group restricted to the site's internal network.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 331
Keysight, EX-1045
Page 331

Mixers and translators may be designed for a variety of purposes. An example is a video mixer that scales the images of individual people in separate video streams and composites them into one video stream to simulate a group scene. Other examples of translation include the connection of a group of hosts speaking only IP/UDP to a group of hosts that understand only ST-II, or the packet-by-packet encoding translation of video streams from individual sources without resynchronization or mixing. Details of the operation of mixers and translators are given in Section 7.

### 2.4 Layered Encodings

Multimedia applications should be able to adjust the transmission rate to match the capacity of the receiver or to adapt to network congestion. Many implementations place the responsibility of rate-adaptivity at the source. This does not work well with multicast transmission because of the conflicting bandwidth requirements of heterogeneous receivers. The result is often a least-common denominator scenario, where the smallest pipe in the network mesh dictates the quality and fidelity of the overall live multimedia "broadcast".

Instead, responsibility for rate-adaptation can be placed at the receivers by combining a layered encoding with a layered transmission system. In the context of RTP over IP multicast, the source can stripe the progressive layers of a hierarchically represented signal across multiple RTP sessions each carried on its own multicast group. Receivers can then adapt to network heterogeneity and control their reception bandwidth by joining only the appropriate subset of the multicast groups.

Details of the use of RTP with layered encodings are given in Sections 6.3.9, 8.3 and 11.

## 3. Definitions

RTP payload: The data transported by RTP in a packet, for example audio samples or compressed video data. The payload format and interpretation are beyond the scope of this document.

RTP packet: A data packet consisting of the fixed RTP header, a possibly empty list of contributing sources (see below), and the payload data. Some underlying protocols may require an encapsulation of the RTP packet to be defined. Typically one packet of the underlying protocol contains a single RTP packet, but several RTP packets MAY be contained if permitted by the encapsulation method (see Section 11).



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 332
Keysight, EX-1045
Page 332

RTCP packet: A control packet consisting of a fixed header part similar to that of RTP data packets, followed by structured elements that vary depending upon the RTCP packet type. The formats are defined in Section 6. Typically, multiple RTCP packets are sent together as a compound RTCP packet in a single packet of the underlying protocol; this is enabled by the length field in the fixed header of each RTCP packet.

Port: The "abstraction that transport protocols use to distinguish among multiple destinations within a given host computer. TCP/IP protocols identify ports using small positive integers." [12] The transport selectors (TSEL) used by the OSI transport layer are equivalent to ports. RTP depends upon the lower-layer protocol to provide some mechanism such as ports to multiplex the RTP and RTCP packets of a session.

Transport address: The combination of a network address and port that identifies a transport-level endpoint, for example an IP address and a UDP port. Packets are transmitted from a source transport address to a destination transport address.

RTP media type: An RTP media type is the collection of payload types which can be carried within a single RTP session. The RTP Profile assigns RTP media types to RTP payload types.

Multimedia session: A set of concurrent RTP sessions among a common group of participants. For example, a videoconference (which is a multimedia session) may contain an audio RTP session and a video RTP session.

RTP session: An association among a set of participants communicating with RTP. A participant may be involved in multiple RTP sessions at the same time. In a multimedia session, each medium is typically carried in a separate RTP session with its own RTCP packets unless the the encoding itself multiplexes multiple media into a single data stream. A participant distinguishes multiple RTP sessions by reception of different sessions using different pairs of destination transport addresses, where a pair of transport addresses comprises one network address plus a pair of ports for RTP and RTCP. All participants in an RTP session may share a common destination transport address pair, as in the case of IP multicast, or the pairs may be different for each participant, as in the case of individual unicast network addresses and port pairs. In the unicast case, a participant may receive from all other participants in the session using the same pair of ports, or may use a distinct pair of ports for each.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 333
Keysight, EX-1045
Page 333

The distinguishing feature of an RTP session is that each maintains a full, separate space of SSRC identifiers (defined next). The set of participants included in one RTP session consists of those that can receive an SSRC identifier transmitted by any one of the participants either in RTP as the SSRC or a CSRC (also defined below) or in RTCP. For example, consider a three-party conference implemented using unicast UDP with each participant receiving from the other two on separate port pairs. If each participant sends RTCP feedback about data received from one other participant only back to that participant, then the conference is composed of three separate point-to-point RTP sessions. If each participant provides RTCP feedback about its reception of one other participant to both of the other participants, then the conference is composed of one multi-party RTP session. The latter case simulates the behavior that would occur with IP multicast communication among the three participants.

The RTP framework allows the variations defined here, but a particular control protocol or application design will usually impose constraints on these variations.

Synchronization source (SSRC): The source of a stream of RTP packets, identified by a 32-bit numeric SSRC identifier carried in the RTP header so as not to be dependent upon the network address. All packets from a synchronization source form part of the same timing and sequence number space, so a receiver groups packets by synchronization source for playback. Examples of synchronization sources include the sender of a stream of packets derived from a signal source such as a microphone or a camera, or an RTP mixer (see below). A synchronization source may change its data format, e.g., audio encoding, over time. The SSRC identifier is a randomly chosen value meant to be globally unique within a particular RTP session (see Section 8). A participant need not use the same SSRC identifier for all the RTP sessions in a multimedia session; the binding of the SSRC identifiers is provided through RTCP (see Section 6.5.1). If a participant generates multiple streams in one RTP session, for example from separate video cameras, each MUST be identified as a different SSRC.

Contributing source (CSRC): A source of a stream of RTP packets that has contributed to the combined stream produced by an RTP mixer (see below). The mixer inserts a list of the SSRC identifiers of the sources that contributed to the generation of a particular packet into the RTP header of that packet. This list is called the CSRC list. An example application is audio conferencing where a mixer indicates all the talkers whose speech



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 334
Keysight, EX-1045
Page 334

was combined to produce the outgoing packet, allowing the receiver to indicate the current talker, even though all the audio packets contain the same SSRC identifier (that of the mixer).

End system: An application that generates the content to be sent in RTP packets and/or consumes the content of received RTP packets. An end system can act as one or more synchronization sources in a particular RTP session, but typically only one.

Mixer: An intermediate system that receives RTP packets from one or more sources, possibly changes the data format, combines the packets in some manner and then forwards a new RTP packet. Since the timing among multiple input sources will not generally be synchronized, the mixer will make timing adjustments among the streams and generate its own timing for the combined stream. Thus, all data packets originating from a mixer will be identified as having the mixer as their synchronization source.

Translator: An intermediate system that forwards RTP packets with their synchronization source identifier intact. Examples of translators include devices that convert encodings without mixing, replicators from multicast to unicast, and application-level filters in firewalls.

Monitor: An application that receives RTCP packets sent by participants in an RTP session, in particular the reception reports, and estimates the current quality of service for distribution monitoring, fault diagnosis and long-term statistics. The monitor function is likely to be built into the application(s) participating in the session, but may also be a separate application that does not otherwise participate and does not send or receive the RTP data packets (since they are on a separate port). These are called third-party monitors. It is also acceptable for a third-party monitor to receive the RTP data packets but not send RTCP packets or otherwise be counted in the session.

Non-RTP means: Protocols and mechanisms that may be needed in addition to RTP to provide a usable service. In particular, for multimedia conferences, a control protocol may distribute multicast addresses and keys for encryption, negotiate the encryption algorithm to be used, and define dynamic mappings between RTP payload type values and the payload formats they represent for formats that do not have a predefined payload type value. Examples of such protocols include the Session Initiation Protocol (SIP) (RFC 3261 [13]), ITU Recommendation H.323 [14] and applications using SDP (RFC 2327 [15]), such as RTSP (RFC 2326 [16]). For simple



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 335
Keysight, EX-1045
Page 335

applications, electronic mail or a conference database may also be used. The specification of such protocols and mechanisms is outside the scope of this document.

## 4. Byte Order, Alignment, and Time Format

All integer fields are carried in network byte order, that is, most significant byte (octet) first. This byte order is commonly known as big-endian. The transmission order is described in detail in [3]. Unless otherwise noted, numeric constants are in decimal (base 10).

All header data is aligned to its natural length, i.e., 16-bit fields are aligned on even offsets, 32-bit fields are aligned at offsets divisible by four, etc. Octets designated as padding have the value zero.

Wallclock time (absolute date and time) is represented using the timestamp format of the Network Time Protocol (NTP), which is in seconds relative to 0h UTC on 1 January 1900 [4]. The full resolution NTP timestamp is a 64-bit unsigned fixed-point number with the integer part in the first 32 bits and the fractional part in the last 32 bits. In some fields where a more compact representation is appropriate, only the middle 32 bits are used; that is, the low 16 bits of the integer part and the high 16 bits of the fractional part. The high 16 bits of the integer part must be determined independently.

An implementation is not required to run the Network Time Protocol in order to use RTP. Other time sources, or none at all, may be used (see the description of the NTP timestamp field in Section 6.4.1). However, running NTP may be useful for synchronizing streams transmitted from separate hosts.

The NTP timestamp will wrap around to zero some time in the year 2036, but for RTP purposes, only differences between pairs of NTP timestamps are used. So long as the pairs of timestamps can be assumed to be within 68 years of each other, using modular arithmetic for subtractions and comparisons makes the wraparound irrelevant.



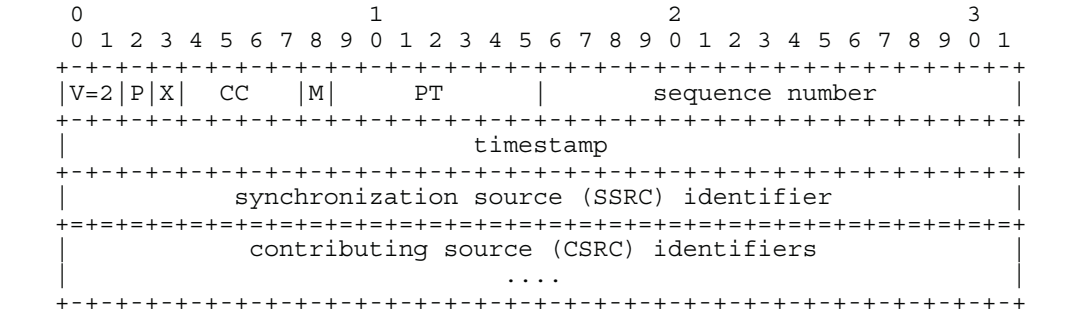
https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 336
Keysight, EX-1045
Page 336

## 5. RTP Data Transfer Protocol

### 5.1 RTP Fixed Header Fields

The RTP header has the following format:




The first twelve octets are present in every RTP packet, while the list of CSRC identifiers is present only when inserted by a mixer. The fields have the following meaning:

version (V): 2 bits
: This field identifies the version of RTP. The version defined by this specification is two (2). (The value 1 is used by the first draft version of RTP and the value 0 is used by the protocol initially implemented in the "vat" audio tool.)

padding (P): 1 bit
: If the padding bit is set, the packet contains one or more additional padding octets at the end which are not part of the payload. The last octet of the padding contains a count of how many padding octets should be ignored, including itself. Padding may be needed by some encryption algorithms with fixed block sizes or for carrying several RTP packets in a lower-layer protocol data unit.

extension (X): 1 bit
: If the extension bit is set, the fixed header MUST be followed by exactly one header extension, with a format defined in Section 5.3.1.

CSRC count (CC): 4 bits
: The CSRC count contains the number of CSRC identifiers that follow the fixed header.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 337
Keysight, EX-1045
Page 337

marker (M): 1 bit
The interpretation of the marker is defined by a profile. It is intended to allow significant events such as frame boundaries to be marked in the packet stream. A profile MAY define additional marker bits or specify that there is no marker bit by changing the number of bits in the payload type field (see Section 5.3).

payload type (PT): 7 bits
This field identifies the format of the RTP payload and determines its interpretation by the application. A profile MAY specify a default static mapping of payload type codes to payload formats. Additional payload type codes MAY be defined dynamically through non-RTP means (see Section 3). A set of default mappings for audio and video is specified in the companion RFC 3551 [1]. An RTP source MAY change the payload type during a session, but this field SHOULD NOT be used for multiplexing separate media streams (see Section 5.2).

A receiver MUST ignore packets with payload types that it does not understand.

sequence number: 16 bits
The sequence number increments by one for each RTP data packet sent, and may be used by the receiver to detect packet loss and to restore packet sequence. The initial value of the sequence number SHOULD be random (unpredictable) to make known-plaintext attacks on encryption more difficult, even if the source itself does not encrypt according to the method in Section 9.1, because the packets may flow through a translator that does. Techniques for choosing unpredictable numbers are discussed in [17].

timestamp: 32 bits
The timestamp reflects the sampling instant of the first octet in the RTP data packet. The sampling instant MUST be derived from a clock that increments monotonically and linearly in time to allow synchronization and jitter calculations (see Section 6.4.1). The resolution of the clock MUST be sufficient for the desired synchronization accuracy and for measuring packet arrival jitter (one tick per video frame is typically not sufficient). The clock frequency is dependent on the format of data carried as payload and is specified statically in the profile or payload format specification that defines the format, or MAY be specified dynamically for payload formats defined through non-RTP means. If RTP packets are generated periodically, the nominal sampling instant as determined from the sampling clock is to be used, not a reading of the system clock. As an example, for fixed-rate audio the timestamp clock would likely increment by one for each sampling period. If an audio application reads blocks covering



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 338
Keysight, EX-1045
Page 338

160 sampling periods from the input device, the timestamp would be increased by 160 for each such block, regardless of whether the block is transmitted in a packet or dropped as silent.

The initial value of the timestamp SHOULD be random, as for the sequence number. Several consecutive RTP packets will have equal timestamps if they are (logically) generated at once, e.g., belong to the same video frame. Consecutive RTP packets MAY contain timestamps that are not monotonic if the data is not transmitted in the order it was sampled, as in the case of MPEG interpolated video frames. (The sequence numbers of the packets as transmitted will still be monotonic.)

RTP timestamps from different media streams may advance at different rates and usually have independent, random offsets. Therefore, although these timestamps are sufficient to reconstruct the timing of a single stream, directly comparing RTP timestamps from different media is not effective for synchronization. Instead, for each medium the RTP timestamp is related to the sampling instant by pairing it with a timestamp from a reference clock (wallclock) that represents the time when the data corresponding to the RTP timestamp was sampled. The reference clock is shared by all media to be synchronized. The timestamp pairs are not transmitted in every data packet, but at a lower rate in RTCP SR packets as described in Section 6.4.

The sampling instant is chosen as the point of reference for the RTP timestamp because it is known to the transmitting endpoint and has a common definition for all media, independent of encoding delays or other processing. The purpose is to allow synchronized presentation of all media sampled at the same time.

Applications transmitting stored data rather than data sampled in real time typically use a virtual presentation timeline derived from wallclock time to determine when the next frame or other unit of each medium in the stored data should be presented. In this case, the RTP timestamp would reflect the presentation time for each unit. That is, the RTP timestamp for each unit would be related to the wallclock time at which the unit becomes current on the virtual presentation timeline. Actual presentation occurs some time later as determined by the receiver.

An example describing live audio narration of prerecorded video illustrates the significance of choosing the sampling instant as the reference point. In this scenario, the video would be presented locally for the narrator to view and would be simultaneously transmitted using RTP. The "sampling instant" of a video frame transmitted in RTP would be established by referencing



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 339
Keysight, EX-1045
Page 339

its timestamp to the wallclock time when that video frame was presented to the narrator. The sampling instant for the audio RTP packets containing the narrator's speech would be established by referencing the same wallclock time when the audio was sampled. The audio and video may even be transmitted by different hosts if the reference clocks on the two hosts are synchronized by some means such as NTP. A receiver can then synchronize presentation of the audio and video packets by relating their RTP timestamps using the timestamp pairs in RTCP SR packets.

SSRC: 32 bits
The SSRC field identifies the synchronization source. This identifier SHOULD be chosen randomly, with the intent that no two synchronization sources within the same RTP session will have the same SSRC identifier. An example algorithm for generating a random identifier is presented in Appendix A.6. Although the probability of multiple sources choosing the same identifier is low, all RTP implementations must be prepared to detect and resolve collisions. Section 8 describes the probability of collision along with a mechanism for resolving collisions and detecting RTP-level forwarding loops based on the uniqueness of the SSRC identifier. If a source changes its source transport address, it must also choose a new SSRC identifier to avoid being interpreted as a looped source (see Section 8.2).

CSRC list: 0 to 15 items, 32 bits each
The CSRC list identifies the contributing sources for the payload contained in this packet. The number of identifiers is given by the CC field. If there are more than 15 contributing sources, only 15 can be identified. CSRC identifiers are inserted by mixers (see Section 7.1), using the SSRC identifiers of contributing sources. For example, for audio packets the SSRC identifiers of all sources that were mixed together to create a packet are listed, allowing correct talker indication at the receiver.

### 5.2 Multiplexing RTP Sessions

For efficient protocol processing, the number of multiplexing points should be minimized, as described in the integrated layer processing design principle [10]. In RTP, multiplexing is provided by the destination transport address (network address and port number) which is different for each RTP session. For example, in a teleconference composed of audio and video media encoded separately, each medium SHOULD be carried in a separate RTP session with its own destination transport address.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 340
Keysight, EX-1045
Page 340

Separate audio and video streams SHOULD NOT be carried in a single RTP session and demultiplexed based on the payload type or SSRC fields. Interleaving packets with different RTP media types but using the same SSRC would introduce several problems:

1. If, say, two audio streams shared the same RTP session and the same SSRC value, and one were to change encodings and thus acquire a different RTP payload type, there would be no general way of identifying which stream had changed encodings.

2. An SSRC is defined to identify a single timing and sequence number space. Interleaving multiple payload types would require different timing spaces if the media clock rates differ and would require different sequence number spaces to tell which payload type suffered packet loss.

3. The RTCP sender and receiver reports (see Section 6.4) can only describe one timing and sequence number space per SSRC and do not carry a payload type field.

4. An RTP mixer would not be able to combine interleaved streams of incompatible media into one stream.

5. Carrying multiple media in one RTP session precludes: the use of different network paths or network resource allocations if appropriate; reception of a subset of the media if desired, for example just audio if video would exceed the available bandwidth; and receiver implementations that use separate processes for the different media, whereas using separate RTP sessions permits either single- or multiple-process implementations.

Using a different SSRC for each medium but sending them in the same RTP session would avoid the first three problems but not the last two.

On the other hand, multiplexing multiple related sources of the same medium in one RTP session using different SSRC values is the norm for multicast sessions. The problems listed above don't apply: an RTP mixer can combine multiple audio sources, for example, and the same treatment is applicable for all of them. It may also be appropriate to multiplex streams of the same medium using different SSRC values in other scenarios where the last two problems do not apply.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 341
Keysight, EX-1045
Page 341

### 5.3 Profile-Specific Modifications to the RTP Header

The existing RTP data packet header is believed to be complete for the set of functions required in common across all the application classes that RTP might support. However, in keeping with the ALF design principle, the header MAY be tailored through modifications or additions defined in a profile specification while still allowing profile-independent monitoring and recording tools to function.

- The marker bit and payload type field carry profile-specific information, but they are allocated in the fixed header since many applications are expected to need them and might otherwise have to add another 32-bit word just to hold them. The octet containing these fields MAY be redefined by a profile to suit different requirements, for example with more or fewer marker bits. If there are any marker bits, one SHOULD be located in the most significant bit of the octet since profile-independent monitors may be able to observe a correlation between packet loss patterns and the marker bit.

- Additional information that is required for a particular payload format, such as a video encoding, SHOULD be carried in the payload section of the packet. This might be in a header that is always present at the start of the payload section, or might be indicated by a reserved value in the data pattern.

- If a particular class of applications needs additional functionality independent of payload format, the profile under which those applications operate SHOULD define additional fixed fields to follow immediately after the SSRC field of the existing fixed header. Those applications will be able to quickly and directly access the additional fields while profile-independent monitors or recorders can still process the RTP packets by interpreting only the first twelve octets.

If it turns out that additional functionality is needed in common across all profiles, then a new version of RTP should be defined to make a permanent change to the fixed header.

#### 5.3.1 RTP Header Extension

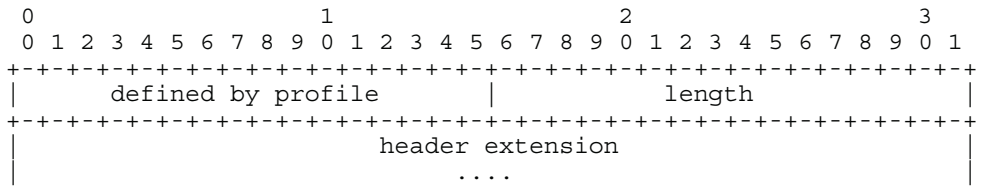
An extension mechanism is provided to allow individual implementations to experiment with new payload-format-independent functions that require additional information to be carried in the RTP data packet header. This mechanism is designed so that the header extension may be ignored by other interoperating implementations that have not been extended.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 342
Keysight, EX-1045
Page 342

Note that this header extension is intended only for limited use. Most potential uses of this mechanism would be better done another way, using the methods described in the previous section. For example, a profile-specific extension to the fixed header is less expensive to process because it is not conditional nor in a variable location. Additional information required for a particular payload format SHOULD NOT use this header extension, but SHOULD be carried in the payload section of the packet.




If the X bit in the RTP header is one, a variable-length header extension MUST be appended to the RTP header, following the CSRC list if present. The header extension contains a 16-bit length field that counts the number of 32-bit words in the extension, excluding the four-octet extension header (therefore zero is a valid length). Only a single extension can be appended to the RTP data header. To allow multiple interoperating implementations to each experiment independently with different header extensions, or to allow a particular implementation to experiment with more than one type of header extension, the first 16 bits of the header extension are left open for distinguishing identifiers or parameters. The format of these 16 bits is to be defined by the profile specification under which the implementations are operating. This RTP specification does not define any header extensions itself.

## 6. RTP Control Protocol -- RTCP

The RTP control protocol (RTCP) is based on the periodic transmission of control packets to all participants in the session, using the same distribution mechanism as the data packets. The underlying protocol MUST provide multiplexing of the data and control packets, for example using separate port numbers with UDP. RTCP performs four functions:

1. The primary function is to provide feedback on the quality of the data distribution. This is an integral part of the RTP's role as a transport protocol and is related to the flow and congestion control functions of other transport protocols (see Section 10 on the requirement for congestion control). The feedback may be directly useful for control of adaptive encodings [18,19], but experiments with IP multicasting have shown that it is also



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 343
Keysight, EX-1045
Page 343

critical to get feedback from the receivers to diagnose faults in the distribution. Sending reception feedback reports to all participants allows one who is observing problems to evaluate whether those problems are local or global. With a distribution mechanism like IP multicast, it is also possible for an entity such as a network service provider who is not otherwise involved in the session to receive the feedback information and act as a third-party monitor to diagnose network problems. This feedback function is performed by the RTCP sender and receiver reports, described below in Section 6.4.

2. RTCP carries a persistent transport-level identifier for an RTP source called the canonical name or CNAME, Section 6.5.1. Since the SSRC identifier may change if a conflict is discovered or a program is restarted, receivers require the CNAME to keep track of each participant. Receivers may also require the CNAME to associate multiple data streams from a given participant in a set of related RTP sessions, for example to synchronize audio and video. Inter-media synchronization also requires the NTP and RTP timestamps included in RTCP packets by data senders.

3. The first two functions require that all participants send RTCP packets, therefore the rate must be controlled in order for RTP to scale up to a large number of participants. By having each participant send its control packets to all the others, each can independently observe the number of participants. This number is used to calculate the rate at which the packets are sent, as explained in Section 6.2.

4. A fourth, OPTIONAL function is to convey minimal session control information, for example participant identification to be displayed in the user interface. This is most likely to be useful in "loosely controlled" sessions where participants enter and leave without membership control or parameter negotiation. RTCP serves as a convenient channel to reach all the participants, but it is not necessarily expected to support all the control communication requirements of an application. A higher-level session control protocol, which is beyond the scope of this document, may be needed.

Functions 1-3 SHOULD be used in all environments, but particularly in the IP multicast environment. RTP application designers SHOULD avoid mechanisms that can only work in unicast mode and will not scale to larger numbers. Transmission of RTCP MAY be controlled separately for senders and receivers, as described in Section 6.2, for cases such as unidirectional links where feedback from receivers is not possible.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 344
Keysight, EX-1045
Page 344

Non-normative note: In the multicast routing approach called Source-Specific Multicast (SSM), there is only one sender per "channel" (a source address, group address pair), and receivers (except for the channel source) cannot use multicast to communicate directly with other channel members. The recommendations here accommodate SSM only through Section 6.2's option of turning off receivers' RTCP entirely. Future work will specify adaptation of RTCP for SSM so that feedback from receivers can be maintained.

### 6.1 RTCP Packet Format

This specification defines several RTCP packet types to carry a variety of control information:

SR: Sender report, for transmission and reception statistics from participants that are active senders

RR: Receiver report, for reception statistics from participants that are not active senders and in combination with SR for active senders reporting on more than 31 sources

SDES: Source description items, including CNAME

BYE: Indicates end of participation

APP: Application-specific functions

Each RTCP packet begins with a fixed part similar to that of RTP data packets, followed by structured elements that MAY be of variable length according to the packet type but MUST end on a 32-bit boundary. The alignment requirement and a length field in the fixed part of each packet are included to make RTCP packets "stackable". Multiple RTCP packets can be concatenated without any intervening separators to form a compound RTCP packet that is sent in a single packet of the lower layer protocol, for example UDP. There is no explicit count of individual RTCP packets in the compound packet since the lower layer protocols are expected to provide an overall length to determine the end of the compound packet.

Each individual RTCP packet in the compound packet may be processed independently with no requirements upon the order or combination of packets. However, in order to perform the functions of the protocol, the following constraints are imposed:



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 345
Keysight, EX-1045
Page 345

- Reception statistics (in SR or RR) should be sent as often as bandwidth constraints will allow to maximize the resolution of the statistics, therefore each periodically transmitted compound RTCP packet MUST include a report packet.

- New receivers need to receive the CNAME for a source as soon as possible to identify the source and to begin associating media for purposes such as lip-sync, so each compound RTCP packet MUST also include the SDES CNAME except when the compound RTCP packet is split for partial encryption as described in Section 9.1.

- The number of packet types that may appear first in the compound packet needs to be limited to increase the number of constant bits in the first word and the probability of successfully validating RTCP packets against misaddressed RTP data packets or other unrelated packets.

Thus, all RTCP packets MUST be sent in a compound packet of at least two individual packets, with the following format:

Encryption prefix: If and only if the compound packet is to be encrypted according to the method in Section 9.1, it MUST be prefixed by a random 32-bit quantity redrawn for every compound packet transmitted. If padding is required for the encryption, it MUST be added to the last packet of the compound packet.

SR or RR: The first RTCP packet in the compound packet MUST always be a report packet to facilitate header validation as described in Appendix A.2. This is true even if no data has been sent or received, in which case an empty RR MUST be sent, and even if the only other RTCP packet in the compound packet is a BYE.

Additional RRs: If the number of sources for which reception statistics are being reported exceeds 31, the number that will fit into one SR or RR packet, then additional RR packets SHOULD follow the initial report packet.

SDES: An SDES packet containing a CNAME item MUST be included in each compound RTCP packet, except as noted in Section 9.1. Other source description items MAY optionally be included if required by a particular application, subject to bandwidth constraints (see Section 6.3.9).

BYE or APP: Other RTCP packet types, including those yet to be defined, MAY follow in any order, except that BYE SHOULD be the last packet sent with a given SSRC/CSRC. Packet types MAY appear more than once.



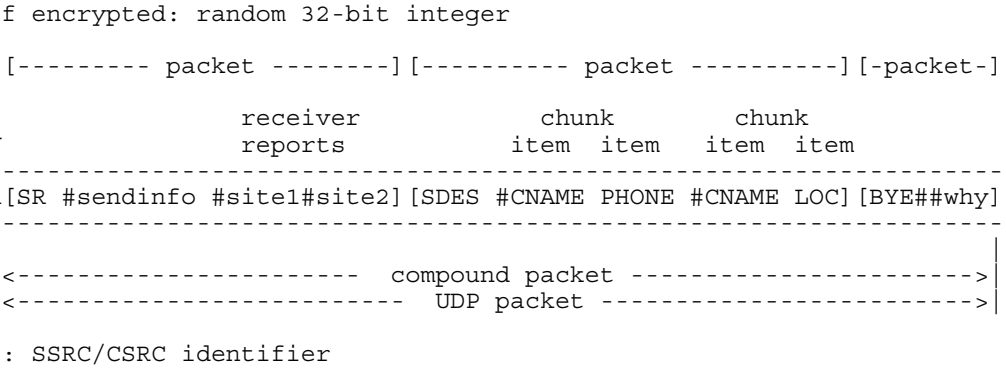
https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 346
Keysight, EX-1045
Page 346

An individual RTP participant SHOULD send only one compound RTCP packet per report interval in order for the RTCP bandwidth per participant to be estimated correctly (see Section 6.2), except when the compound RTCP packet is split for partial encryption as described in Section 9.1. If there are too many sources to fit all the necessary RR packets into one compound RTCP packet without exceeding the maximum transmission unit (MTU) of the network path, then only the subset that will fit into one MTU SHOULD be included in each interval. The subsets SHOULD be selected round-robin across multiple intervals so that all sources are reported.

It is RECOMMENDED that translators and mixers combine individual RTCP packets from the multiple sources they are forwarding into one compound packet whenever feasible in order to amortize the packet overhead (see Section 7). An example RTCP compound packet as might be produced by a mixer is shown in Fig. 1. If the overall length of a compound packet would exceed the MTU of the network path, it SHOULD be segmented into multiple shorter compound packets to be transmitted in separate packets of the underlying protocol. This does not impair the RTCP bandwidth estimation because each compound packet represents at least one distinct participant. Note that each of the compound packets MUST begin with an SR or RR packet.

An implementation SHOULD ignore incoming RTCP packets with types unknown to it. Additional RTCP packet types may be registered with the Internet Assigned Numbers Authority (IANA) as described in Section 15.




#: SSRC/CSRC identifier

Figure 1: Example of an RTCP compound packet



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 347
Keysight, EX-1045
Page 347

### 6.2 RTCP Transmission Interval

RTP is designed to allow an application to scale automatically over session sizes ranging from a few participants to thousands. For example, in an audio conference the data traffic is inherently self-limiting because only one or two people will speak at a time, so with multicast distribution the data rate on any given link remains relatively constant independent of the number of participants. However, the control traffic is not self-limiting. If the reception reports from each participant were sent at a constant rate, the control traffic would grow linearly with the number of participants. Therefore, the rate must be scaled down by dynamically calculating the interval between RTCP packet transmissions.

For each session, it is assumed that the data traffic is subject to an aggregate limit called the "session bandwidth" to be divided among the participants. This bandwidth might be reserved and the limit enforced by the network. If there is no reservation, there may be other constraints, depending on the environment, that establish the "reasonable" maximum for the session to use, and that would be the session bandwidth. The session bandwidth may be chosen based on some cost or a priori knowledge of the available network bandwidth for the session. It is somewhat independent of the media encoding, but the encoding choice may be limited by the session bandwidth. Often, the session bandwidth is the sum of the nominal bandwidths of the senders expected to be concurrently active. For teleconference audio, this number would typically be one sender's bandwidth. For layered encodings, each layer is a separate RTP session with its own session bandwidth parameter.

The session bandwidth parameter is expected to be supplied by a session management application when it invokes a media application, but media applications MAY set a default based on the single-sender data bandwidth for the encoding selected for the session. The application MAY also enforce bandwidth limits based on multicast scope rules or other criteria. All participants MUST use the same value for the session bandwidth so that the same RTCP interval will be calculated.

Bandwidth calculations for control and data traffic include lower-layer transport and network protocols (e.g., UDP and IP) since that is what the resource reservation system would need to know. The application can also be expected to know which of these protocols are in use. Link level headers are not included in the calculation since the packet will be encapsulated with different link level headers as it travels.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 348
Keysight, EX-1045
Page 348

The control traffic should be limited to a small and known fraction of the session bandwidth: small so that the primary function of the transport protocol to carry data is not impaired; known so that the control traffic can be included in the bandwidth specification given to a resource reservation protocol, and so that each participant can independently calculate its share. The control traffic bandwidth is in addition to the session bandwidth for the data traffic. It is RECOMMENDED that the fraction of the session bandwidth added for RTCP be fixed at 5%. It is also RECOMMENDED that 1/4 of the RTCP bandwidth be dedicated to participants that are sending data so that in sessions with a large number of receivers but a small number of senders, newly joining participants will more quickly receive the CNAME for the sending sites. When the proportion of senders is greater than 1/4 of the participants, the senders get their proportion of the full RTCP bandwidth. While the values of these and other constants in the interval calculation are not critical, all participants in the session MUST use the same values so the same interval will be calculated. Therefore, these constants SHOULD be fixed for a particular profile.

A profile MAY specify that the control traffic bandwidth may be a separate parameter of the session rather than a strict percentage of the session bandwidth. Using a separate parameter allows rate-adaptive applications to set an RTCP bandwidth consistent with a "typical" data bandwidth that is lower than the maximum bandwidth specified by the session bandwidth parameter.

The profile MAY further specify that the control traffic bandwidth may be divided into two separate session parameters for those participants which are active data senders and those which are not; let us call the parameters S and R. Following the recommendation that 1/4 of the RTCP bandwidth be dedicated to data senders, the RECOMMENDED default values for these two parameters would be 1.25% and 3.75%, respectively. When the proportion of senders is greater than S/(S+R) of the participants, the senders get their proportion of the sum of these parameters. Using two parameters allows RTCP reception reports to be turned off entirely for a particular session by setting the RTCP bandwidth for non-data-senders to zero while keeping the RTCP bandwidth for data senders non-zero so that sender reports can still be sent for inter-media synchronization. Turning off RTCP reception reports is NOT RECOMMENDED because they are needed for the functions listed at the beginning of Section 6, particularly reception quality feedback and congestion control. However, doing so may be appropriate for systems operating on unidirectional links or for sessions that don't require feedback on the quality of reception or liveness of receivers and that have other means to avoid congestion.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 349
Keysight, EX-1045
Page 349

The calculated interval between transmissions of compound RTCP packets SHOULD also have a lower bound to avoid having bursts of packets exceed the allowed bandwidth when the number of participants is small and the traffic isn't smoothed according to the law of large numbers. It also keeps the report interval from becoming too small during transient outages like a network partition such that adaptation is delayed when the partition heals. At application startup, a delay SHOULD be imposed before the first compound RTCP packet is sent to allow time for RTCP packets to be received from other participants so the report interval will converge to the correct value more quickly. This delay MAY be set to half the minimum interval to allow quicker notification that the new participant is present. The RECOMMENDED value for a fixed minimum interval is 5 seconds.

An implementation MAY scale the minimum RTCP interval to a smaller value inversely proportional to the session bandwidth parameter with the following limitations:

- For multicast sessions, only active data senders MAY use the reduced minimum value to calculate the interval for transmission of compound RTCP packets.

- For unicast sessions, the reduced value MAY be used by participants that are not active data senders as well, and the delay before sending the initial compound RTCP packet MAY be zero.

- For all sessions, the fixed minimum SHOULD be used when calculating the participant timeout interval (see Section 6.3.5) so that implementations which do not use the reduced value for transmitting RTCP packets are not timed out by other participants prematurely.

- The RECOMMENDED value for the reduced minimum in seconds is 360 divided by the session bandwidth in kilobits/second. This minimum is smaller than 5 seconds for bandwidths greater than 72 kb/s.

The algorithm described in Section 6.3 and Appendix A.7 was designed to meet the goals outlined in this section. It calculates the interval between sending compound RTCP packets to divide the allowed control traffic bandwidth among the participants. This allows an application to provide fast response for small sessions where, for example, identification of all participants is important, yet automatically adapt to large sessions. The algorithm incorporates the following characteristics:



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 350
Keysight, EX-1045
Page 350

- The calculated interval between RTCP packets scales linearly with the number of members in the group. It is this linear factor which allows for a constant amount of control traffic when summed across all members.

- The interval between RTCP packets is varied randomly over the range [0.5,1.5] times the calculated interval to avoid unintended synchronization of all participants [20]. The first RTCP packet sent after joining a session is also delayed by a random variation of half the minimum RTCP interval.

- A dynamic estimate of the average compound RTCP packet size is calculated, including all those packets received and sent, to automatically adapt to changes in the amount of control information carried.

- Since the calculated interval is dependent on the number of observed group members, there may be undesirable startup effects when a new user joins an existing session, or many users simultaneously join a new session. These new users will initially have incorrect estimates of the group membership, and thus their RTCP transmission interval will be too short. This problem can be significant if many users join the session simultaneously. To deal with this, an algorithm called "timer reconsideration" is employed. This algorithm implements a simple back-off mechanism which causes users to hold back RTCP packet transmission if the group sizes are increasing.

- When users leave a session, either with a BYE or by timeout, the group membership decreases, and thus the calculated interval should decrease. A "reverse reconsideration" algorithm is used to allow members to more quickly reduce their intervals in response to group membership decreases.

- BYE packets are given different treatment than other RTCP packets. When a user leaves a group, and wishes to send a BYE packet, it may do so before its next scheduled RTCP packet. However, transmission of BYEs follows a back-off algorithm which avoids floods of BYE packets should a large number of members simultaneously leave the session.

This algorithm may be used for sessions in which all participants are allowed to send. In that case, the session bandwidth parameter is the product of the individual sender's bandwidth times the number of participants, and the RTCP bandwidth is 5% of that.

Details of the algorithm's operation are given in the sections that follow. Appendix A.7 gives an example implementation.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 351
Keysight, EX-1045
Page 351

### 6.2.1 Maintaining the Number of Session Members

Calculation of the RTCP packet interval depends upon an estimate of the number of sites participating in the session. New sites are added to the count when they are heard, and an entry for each SHOULD be created in a table indexed by the SSRC or CSRC identifier (see Section 8.2) to keep track of them. New entries MAY be considered not valid until multiple packets carrying the new SSRC have been received (see Appendix A.1), or until an SDES RTCP packet containing a CNAME for that SSRC has been received. Entries MAY be deleted from the table when an RTCP BYE packet with the corresponding SSRC identifier is received, except that some straggler data packets might arrive after the BYE and cause the entry to be recreated. Instead, the entry SHOULD be marked as having received a BYE and then deleted after an appropriate delay.

A participant MAY mark another site inactive, or delete it if not yet valid, if no RTP or RTCP packet has been received for a small number of RTCP report intervals (5 is RECOMMENDED). This provides some robustness against packet loss. All sites must have the same value for this multiplier and must calculate roughly the same value for the RTCP report interval in order for this timeout to work properly. Therefore, this multiplier SHOULD be fixed for a particular profile.

For sessions with a very large number of participants, it may be impractical to maintain a table to store the SSRC identifier and state information for all of them. An implementation MAY use SSRC sampling, as described in [21], to reduce the storage requirements. An implementation MAY use any other algorithm with similar performance. A key requirement is that any algorithm considered SHOULD NOT substantially underestimate the group size, although it MAY overestimate.

## 6.3 RTCP Packet Send and Receive Rules

The rules for how to send, and what to do when receiving an RTCP packet are outlined here. An implementation that allows operation in a multicast environment or a multipoint unicast environment MUST meet the requirements in Section 6.2. Such an implementation MAY use the algorithm defined in this section to meet those requirements, or MAY use some other algorithm so long as it provides equivalent or better performance. An implementation which is constrained to two-party unicast operation SHOULD still use randomization of the RTCP transmission interval to avoid unintended synchronization of multiple instances operating in the same environment, but MAY omit the "timer reconsideration" and "reverse reconsideration" algorithms in Sections 6.3.3, 6.3.6 and 6.3.7.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 352
Keysight, EX-1045
Page 352

To execute these rules, a session participant must maintain several pieces of state:

tp: the last time an RTCP packet was transmitted;

tc: the current time;

tn: the next scheduled transmission time of an RTCP packet;

pmembers: the estimated number of session members at the time tn was last recomputed;

members: the most current estimate for the number of session members;

senders: the most current estimate for the number of senders in the session;

rtcp_bw: The target RTCP bandwidth, i.e., the total bandwidth that will be used for RTCP packets by all members of this session, in octets per second. This will be a specified fraction of the "session bandwidth" parameter supplied to the application at startup.

we_sent: Flag that is true if the application has sent data since the 2nd previous RTCP report was transmitted.

avg_rtcp_size: The average compound RTCP packet size, in octets, over all RTCP packets sent and received by this participant. The size includes lower-layer transport and network protocol headers (e.g., UDP and IP) as explained in Section 6.2.

initial: Flag that is true if the application has not yet sent an RTCP packet.

Many of these rules make use of the "calculated interval" between packet transmissions. This interval is described in the following section.

### 6.3.1 Computing the RTCP Transmission Interval

To maintain scalability, the average interval between packets from a session participant should scale with the group size. This interval is called the calculated interval. It is obtained by combining a number of the pieces of state described above. The calculated interval T is then determined as follows:



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 353
Keysight, EX-1045
Page 353

1. If the number of senders is less than or equal to 25% of the membership (members), the interval depends on whether the participant is a sender or not (based on the value of we_sent). If the participant is a sender (we_sent true), the constant C is set to the average RTCP packet size (avg_rtcp_size) divided by 25% of the RTCP bandwidth (rtcp_bw), and the constant n is set to the number of senders. If we_sent is not true, the constant C is set to the average RTCP packet size divided by 75% of the RTCP bandwidth. The constant n is set to the number of receivers (members - senders). If the number of senders is greater than 25%, senders and receivers are treated together. The constant C is set to the average RTCP packet size divided by the total RTCP bandwidth and n is set to the total number of members. As stated in Section 6.2, an RTP profile MAY specify that the RTCP bandwidth may be explicitly defined by two separate parameters (call them S and R) for those participants which are senders and those which are not. In that case, the 25% fraction becomes S/(S+R) and the 75% fraction becomes R/(S+R). Note that if R is zero, the percentage of senders is never greater than S/(S+R), and the implementation must avoid division by zero.

2. If the participant has not yet sent an RTCP packet (the variable initial is true), the constant Tmin is set to 2.5 seconds, else it is set to 5 seconds.

3. The deterministic calculated interval Td is set to max(Tmin, n*C).

4. The calculated interval T is set to a number uniformly distributed between 0.5 and 1.5 times the deterministic calculated interval.

5. The resulting value of T is divided by e-3/2=1.21828 to compensate for the fact that the timer reconsideration algorithm converges to a value of the RTCP bandwidth below the intended average.

This procedure results in an interval which is random, but which, on average, gives at least 25% of the RTCP bandwidth to senders and the rest to receivers. If the senders constitute more than one quarter of the membership, this procedure splits the bandwidth equally among all participants, on average.

### 6.3.2 Initialization

Upon joining the session, the participant initializes tp to 0, tc to 0, senders to 0, pmembers to 1, members to 1, we_sent to false, rtcp_bw to the specified fraction of the session bandwidth, initial to true, and avg_rtcp_size to the probable size of the first RTCP packet that the application will later construct. The calculated interval T is then computed, and the first packet is scheduled for



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 354
Keysight, EX-1045
Page 354

time tn = T. This means that a transmission timer is set which expires at time T. Note that an application MAY use any desired approach for implementing this timer.

The participant adds its own SSRC to the member table.

### 6.3.3 Receiving an RTP or Non-BYE RTCP Packet

When an RTP or RTCP packet is received from a participant whose SSRC is not in the member table, the SSRC is added to the table, and the value for members is updated once the participant has been validated as described in Section 6.2.1. The same processing occurs for each CSRC in a validated RTP packet.

When an RTP packet is received from a participant whose SSRC is not in the sender table, the SSRC is added to the table, and the value for senders is updated.

For each compound RTCP packet received, the value of avg_rtcp_size is updated:

```
avg_rtcp_size = (1/16) * packet_size + (15/16) * avg_rtcp_size
```

where packet_size is the size of the RTCP packet just received.

### 6.3.4 Receiving an RTCP BYE Packet

Except as described in Section 6.3.7 for the case when an RTCP BYE is to be transmitted, if the received packet is an RTCP BYE packet, the SSRC is checked against the member table. If present, the entry is removed from the table, and the value for members is updated. The SSRC is then checked against the sender table. If present, the entry is removed from the table, and the value for senders is updated.

Furthermore, to make the transmission rate of RTCP packets more adaptive to changes in group membership, the following "reverse reconsideration" algorithm SHOULD be executed when a BYE packet is received that reduces members to a value less than pmembers:

- The value for tn is updated according to the following formula:

  ```
  tn = tc + (members/pmembers) * (tn - tc)
  ```

- The value for tp is updated according the following formula:

  ```
  tp = tc - (members/pmembers) * (tc - tp).
  ```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 355
Keysight, EX-1045
Page 355

- The next RTCP packet is rescheduled for transmission at time tn, which is now earlier.

- The value of pmembers is set equal to members.

This algorithm does not prevent the group size estimate from incorrectly dropping to zero for a short time due to premature timeouts when most participants of a large session leave at once but some remain. The algorithm does make the estimate return to the correct value more rapidly. This situation is unusual enough and the consequences are sufficiently harmless that this problem is deemed only a secondary concern.

### 6.3.5 Timing Out an SSRC

At occasional intervals, the participant MUST check to see if any of the other participants time out. To do this, the participant computes the deterministic (without the randomization factor) calculated interval Td for a receiver, that is, with we_sent false. Any other session member who has not sent an RTP or RTCP packet since time tc - MTd (M is the timeout multiplier, and defaults to 5) is timed out. This means that its SSRC is removed from the member list, and members is updated. A similar check is performed on the sender list. Any member on the sender list who has not sent an RTP packet since time tc - 2T (within the last two RTCP report intervals) is removed from the sender list, and senders is updated.

If any members time out, the reverse reconsideration algorithm described in Section 6.3.4 SHOULD be performed.

The participant MUST perform this check at least once per RTCP transmission interval.

### 6.3.6 Expiration of Transmission Timer

When the packet transmission timer expires, the participant performs the following operations:

- The transmission interval T is computed as described in Section 6.3.1, including the randomization factor.

- If tp + T is less than or equal to tc, an RTCP packet is transmitted. tp is set to tc, then another value for T is calculated as in the previous step and tn is set to tc + T. The transmission timer is set to expire again at time tn. If tp + T is greater than tc, tn is set to tp + T. No RTCP packet is transmitted. The transmission timer is set to expire at time tn.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 356
Keysight, EX-1045
Page 356

- pmembers is set to members.

If an RTCP packet is transmitted, the value of initial is set to FALSE. Furthermore, the value of avg_rtcp_size is updated:

```
avg_rtcp_size = (1/16) * packet_size + (15/16) * avg_rtcp_size
```

where packet_size is the size of the RTCP packet just transmitted.

### 6.3.7 Transmitting a BYE Packet

When a participant wishes to leave a session, a BYE packet is transmitted to inform the other participants of the event. In order to avoid a flood of BYE packets when many participants leave the system, a participant MUST execute the following algorithm if the number of members is more than 50 when the participant chooses to leave. This algorithm usurps the normal role of the members variable to count BYE packets instead:

- When the participant decides to leave the system, tp is reset to tc, the current time, members and pmembers are initialized to 1, initial is set to 1, we_sent is set to false, senders is set to 0, and avg_rtcp_size is set to the size of the compound BYE packet. The calculated interval T is computed. The BYE packet is then scheduled for time tn = tc + T.

- Every time a BYE packet from another participant is received, members is incremented by 1 regardless of whether that participant exists in the member table or not, and when SSRC sampling is in use, regardless of whether or not the BYE SSRC would be included in the sample. members is NOT incremented when other RTCP packets or RTP packets are received, but only for BYE packets. Similarly, avg_rtcp_size is updated only for received BYE packets. senders is NOT updated when RTP packets arrive; it remains 0.

- Transmission of the BYE packet then follows the rules for transmitting a regular RTCP packet, as above.

This allows BYE packets to be sent right away, yet controls their total bandwidth usage. In the worst case, this could cause RTCP control packets to use twice the bandwidth as normal (10%) -- 5% for non-BYE RTCP packets and 5% for BYE.

A participant that does not want to wait for the above mechanism to allow transmission of a BYE packet MAY leave the group without sending a BYE at all. That participant will eventually be timed out by the other group members.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 357
Keysight, EX-1045
Page 357

If the group size estimate members is less than 50 when the participant decides to leave, the participant MAY send a BYE packet immediately. Alternatively, the participant MAY choose to execute the above BYE backoff algorithm.

In either case, a participant which never sent an RTP or RTCP packet MUST NOT send a BYE packet when they leave the group.

### 6.3.8 Updating we_sent

The variable we_sent contains true if the participant has sent an RTP packet recently, false otherwise. This determination is made by using the same mechanisms as for managing the set of other participants listed in the senders table. If the participant sends an RTP packet when we_sent is false, it adds itself to the sender table and sets we_sent to true. The reverse reconsideration algorithm described in Section 6.3.4 SHOULD be performed to possibly reduce the delay before sending an SR packet. Every time another RTP packet is sent, the time of transmission of that packet is maintained in the table. The normal sender timeout algorithm is then applied to the participant -- if an RTP packet has not been transmitted since time tc - 2T, the participant removes itself from the sender table, decrements the sender count, and sets we_sent to false.

### 6.3.9 Allocation of Source Description Bandwidth

This specification defines several source description (SDES) items in addition to the mandatory CNAME item, such as NAME (personal name) and EMAIL (email address). It also provides a means to define new application-specific RTCP packet types. Applications should exercise caution in allocating control bandwidth to this additional information because it will slow down the rate at which reception reports and CNAME are sent, thus impairing the performance of the protocol. It is RECOMMENDED that no more than 20% of the RTCP bandwidth allocated to a single participant be used to carry the additional information. Furthermore, it is not intended that all SDES items will be included in every application. Those that are included SHOULD be assigned a fraction of the bandwidth according to their utility. Rather than estimate these fractions dynamically, it is recommended that the percentages be translated statically into report interval counts based on the typical length of an item.

For example, an application may be designed to send only CNAME, NAME and EMAIL and not any others. NAME might be given much higher priority than EMAIL because the NAME would be displayed continuously in the application's user interface, whereas EMAIL would be displayed only when requested. At every RTCP interval, an RR packet and an SDES packet with the CNAME item would be sent. For a small session



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 358
Keysight, EX-1045
Page 358

operating at the minimum interval, that would be every 5 seconds on the average. Every third interval (15 seconds), one extra item would be included in the SDES packet. Seven out of eight times this would be the NAME item, and every eighth time (2 minutes) it would be the EMAIL item.

When multiple applications operate in concert using cross-application binding through a common CNAME for each participant, for example in a multimedia conference composed of an RTP session for each medium, the additional SDES information MAY be sent in only one RTP session. The other sessions would carry only the CNAME item. In particular, this approach should be applied to the multiple sessions of a layered encoding scheme (see Section 2.4).

### 6.4 Sender and Receiver Reports

RTP receivers provide reception quality feedback using RTCP report packets which may take one of two forms depending upon whether or not the receiver is also a sender. The only difference between the sender report (SR) and receiver report (RR) forms, besides the packet type code, is that the sender report includes a 20-byte sender information section for use by active senders. The SR is issued if a site has sent any data packets during the interval since issuing the last report or the previous one, otherwise the RR is issued.

Both the SR and RR forms include zero or more reception report blocks, one for each of the synchronization sources from which this receiver has received RTP data packets since the last report. Reports are not issued for contributing sources listed in the CSRC list. Each reception report block provides statistics about the data received from the particular source indicated in that block. Since a maximum of 31 reception report blocks will fit in an SR or RR packet, additional RR packets SHOULD be stacked after the initial SR or RR packet as needed to contain the reception reports for all sources heard during the interval since the last report. If there are too many sources to fit all the necessary RR packets into one compound RTCP packet without exceeding the MTU of the network path, then only the subset that will fit into one MTU SHOULD be included in each interval. The subsets SHOULD be selected round-robin across multiple intervals so that all sources are reported.

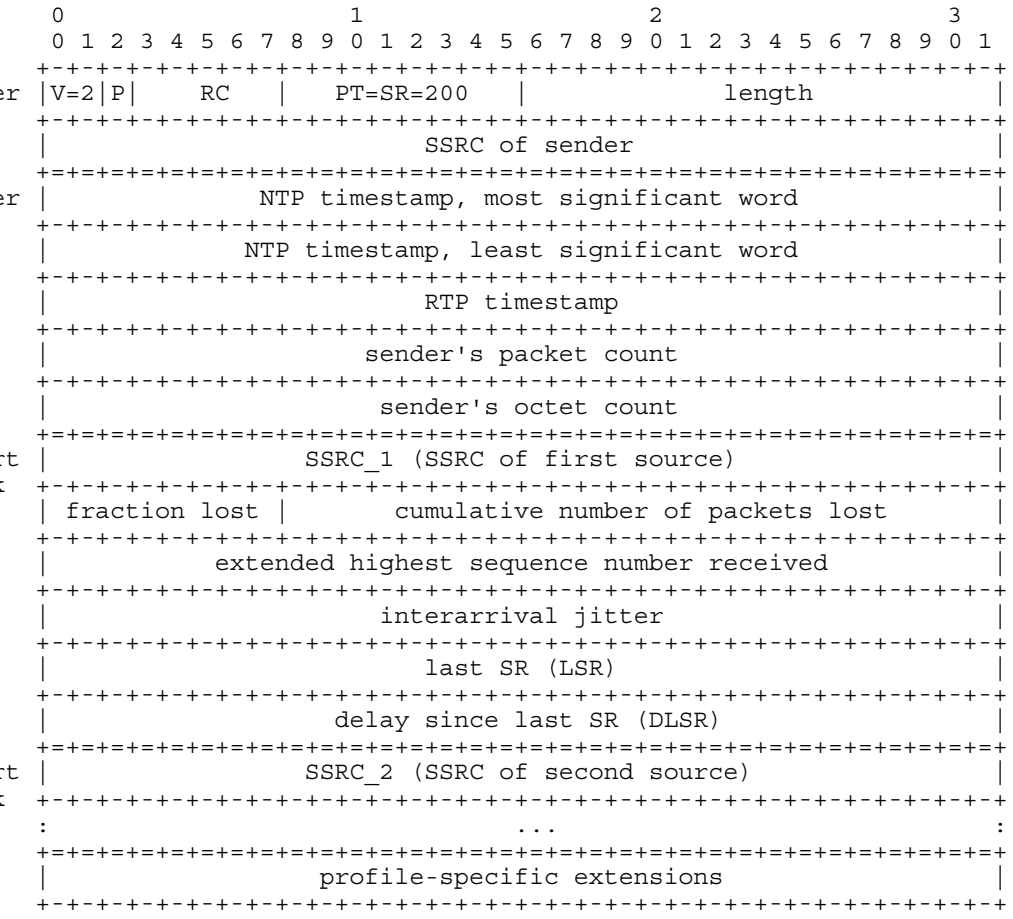
The next sections define the formats of the two reports, how they may be extended in a profile-specific manner if an application requires additional feedback information, and how the reports may be used. Details of reception reporting by translators and mixers is given in Section 7.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 359
Keysight, EX-1045
Page 359

### 6.4.1 SR: Sender Report RTCP Packet




The sender report packet consists of three sections, possibly followed by a fourth profile-specific extension section if defined. The first section, the header, is 8 octets long. The fields have the following meaning:

version (V): 2 bits
Identifies the version of RTP, which is the same in RTCP packets as in RTP data packets. The version defined by this specification is two (2).



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 360
Keysight, EX-1045
Page 360

padding (P): 1 bit
If the padding bit is set, this individual RTCP packet contains some additional padding octets at the end which are not part of the control information but are included in the length field. The last octet of the padding is a count of how many padding octets should be ignored, including itself (it will be a multiple of four). Padding may be needed by some encryption algorithms with fixed block sizes. In a compound RTCP packet, padding is only required on one individual packet because the compound packet is encrypted as a whole for the method in Section 9.1. Thus, padding MUST only be added to the last individual packet, and if padding is added to that packet, the padding bit MUST be set only on that packet. This convention aids the header validity checks described in Appendix A.2 and allows detection of packets from some early implementations that incorrectly set the padding bit on the first individual packet and add padding to the last individual packet.

reception report count (RC): 5 bits
The number of reception report blocks contained in this packet. A value of zero is valid.

packet type (PT): 8 bits
Contains the constant 200 to identify this as an RTCP SR packet.

length: 16 bits
The length of this RTCP packet in 32-bit words minus one, including the header and any padding. (The offset of one makes zero a valid length and avoids a possible infinite loop in scanning a compound RTCP packet, while counting 32-bit words avoids a validity check for a multiple of 4.)

SSRC: 32 bits
The synchronization source identifier for the originator of this SR packet.

The second section, the sender information, is 20 octets long and is present in every sender report packet. It summarizes the data transmissions from this sender. The fields have the following meaning:

NTP timestamp: 64 bits
Indicates the wallclock time (see Section 4) when this report was sent so that it may be used in combination with timestamps returned in reception reports from other receivers to measure round-trip propagation to those receivers. Receivers should expect that the measurement accuracy of the timestamp may be limited to far less than the resolution of the NTP timestamp. The measurement uncertainty of the timestamp is not indicated as it



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 361
Keysight, EX-1045
Page 361

may not be known. On a system that has no notion of wallclock time but does have some system-specific clock such as "system uptime", a sender MAY use that clock as a reference to calculate relative NTP timestamps. It is important to choose a commonly used clock so that if separate implementations are used to produce the individual streams of a multimedia session, all implementations will use the same clock. Until the year 2036, relative and absolute timestamps will differ in the high bit so (invalid) comparisons will show a large difference; by then one hopes relative timestamps will no longer be needed. A sender that has no notion of wallclock or elapsed time MAY set the NTP timestamp to zero.

RTP timestamp: 32 bits
Corresponds to the same time as the NTP timestamp (above), but in the same units and with the same random offset as the RTP timestamps in data packets. This correspondence may be used for intra- and inter-media synchronization for sources whose NTP timestamps are synchronized, and may be used by media-independent receivers to estimate the nominal RTP clock frequency. Note that in most cases this timestamp will not be equal to the RTP timestamp in any adjacent data packet. Rather, it MUST be calculated from the corresponding NTP timestamp using the relationship between the RTP timestamp counter and real time as maintained by periodically checking the wallclock time at a sampling instant.

sender's packet count: 32 bits
The total number of RTP data packets transmitted by the sender since starting transmission up until the time this SR packet was generated. The count SHOULD be reset if the sender changes its SSRC identifier.

sender's octet count: 32 bits
The total number of payload octets (i.e., not including header or padding) transmitted in RTP data packets by the sender since starting transmission up until the time this SR packet was generated. The count SHOULD be reset if the sender changes its SSRC identifier. This field can be used to estimate the average payload data rate.

The third section contains zero or more reception report blocks depending on the number of other sources heard by this sender since the last report. Each reception report block conveys statistics on the reception of RTP packets from a single synchronization source. Receivers SHOULD NOT carry over statistics when a source changes its SSRC identifier due to a collision. These statistics are:



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 362
Keysight, EX-1045
Page 362

SSRC_n (source identifier): 32 bits
The SSRC identifier of the source to which the information in this reception report block pertains.

fraction lost: 8 bits
The fraction of RTP data packets from source SSRC_n lost since the previous SR or RR packet was sent, expressed as a fixed point number with the binary point at the left edge of the field. (That is equivalent to taking the integer part after multiplying the loss fraction by 256.) This fraction is defined to be the number of packets lost divided by the number of packets expected, as defined in the next paragraph. An implementation is shown in Appendix A.3. If the loss is negative due to duplicates, the fraction lost is set to zero. Note that a receiver cannot tell whether any packets were lost after the last one received, and that there will be no reception report block issued for a source if all packets from that source sent during the last reporting interval have been lost.

cumulative number of packets lost: 24 bits
The total number of RTP data packets from source SSRC_n that have been lost since the beginning of reception. This number is defined to be the number of packets expected less the number of packets actually received, where the number of packets received includes any which are late or duplicates. Thus, packets that arrive late are not counted as lost, and the loss may be negative if there are duplicates. The number of packets expected is defined to be the extended last sequence number received, as defined next, less the initial sequence number received. This may be calculated as shown in Appendix A.3.

extended highest sequence number received: 32 bits
The low 16 bits contain the highest sequence number received in an RTP data packet from source SSRC_n, and the most significant 16 bits extend that sequence number with the corresponding count of sequence number cycles, which may be maintained according to the algorithm in Appendix A.1. Note that different receivers within the same session will generate different extensions to the sequence number if their start times differ significantly.

interarrival jitter: 32 bits
An estimate of the statistical variance of the RTP data packet interarrival time, measured in timestamp units and expressed as an unsigned integer. The interarrival jitter J is defined to be the mean deviation (smoothed absolute value) of the difference D in packet spacing at the receiver compared to the sender for a pair of packets. As shown in the equation below, this is equivalent to the difference in the "relative transit time" for the two packets;



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 363
Keysight, EX-1045
Page 363

the relative transit time is the difference between a packet's RTP timestamp and the receiver's clock at the time of arrival, measured in the same units.

If Si is the RTP timestamp from packet i, and Ri is the time of arrival in RTP timestamp units for packet i, then for two packets i and j, D may be expressed as

$$D(i,j) = (Rj - Ri) - (Sj - Si) = (Rj - Sj) - (Ri - Si)$$

The interarrival jitter SHOULD be calculated continuously as each data packet i is received from source SSRC_n, using this difference D for that packet and the previous packet i-1 in order of arrival (not necessarily in sequence), according to the formula

$$J(i) = J(i-1) + (|D(i-1,i)| - J(i-1))/16$$

Whenever a reception report is issued, the current value of J is sampled.

The jitter calculation MUST conform to the formula specified here in order to allow profile-independent monitors to make valid interpretations of reports coming from different implementations. This algorithm is the optimal first-order estimator and the gain parameter 1/16 gives a good noise reduction ratio while maintaining a reasonable rate of convergence [22]. A sample implementation is shown in Appendix A.8. See Section 6.4.4 for a discussion of the effects of varying packet duration and delay before transmission.

last SR timestamp (LSR): 32 bits
The middle 32 bits out of 64 in the NTP timestamp (as explained in Section 4) received as part of the most recent RTCP sender report (SR) packet from source SSRC_n. If no SR has been received yet, the field is set to zero.

delay since last SR (DLSR): 32 bits
The delay, expressed in units of 1/65536 seconds, between receiving the last SR packet from source SSRC_n and sending this reception report block. If no SR packet has been received yet from SSRC_n, the DLSR field is set to zero.

Let SSRC_r denote the receiver issuing this receiver report. Source SSRC_n can compute the round-trip propagation delay to SSRC_r by recording the time A when this reception report block is received. It calculates the total round-trip time A-LSR using the last SR timestamp (LSR) field, and then subtracting this field to leave the round-trip propagation delay as (A - LSR - DLSR). This



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 364
Keysight, EX-1045
Page 364

is illustrated in Fig. 2. Times are shown in both a hexadecimal representation of the 32-bit fields and the equivalent floating-point decimal representation. Colons indicate a 32-bit field divided into a 16-bit integer part and 16-bit fraction part.

This may be used as an approximate measure of distance to cluster receivers, although some links have very asymmetric delays.

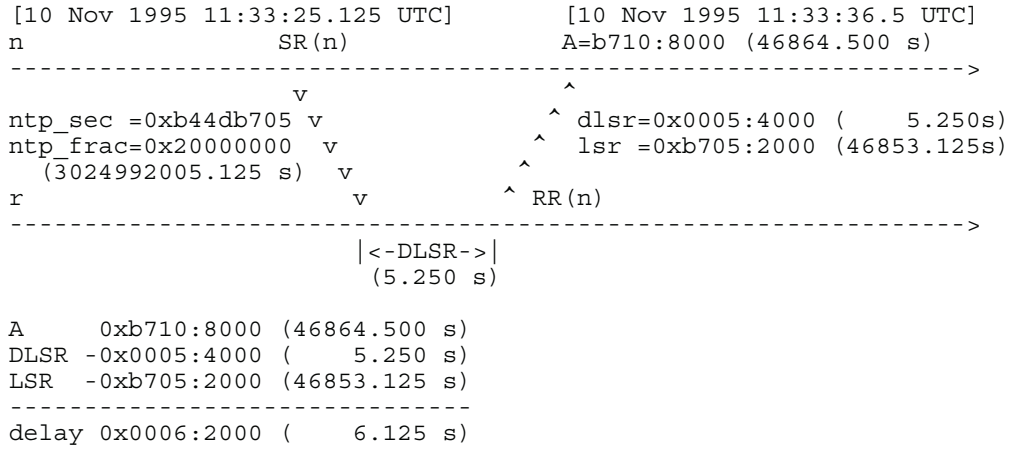


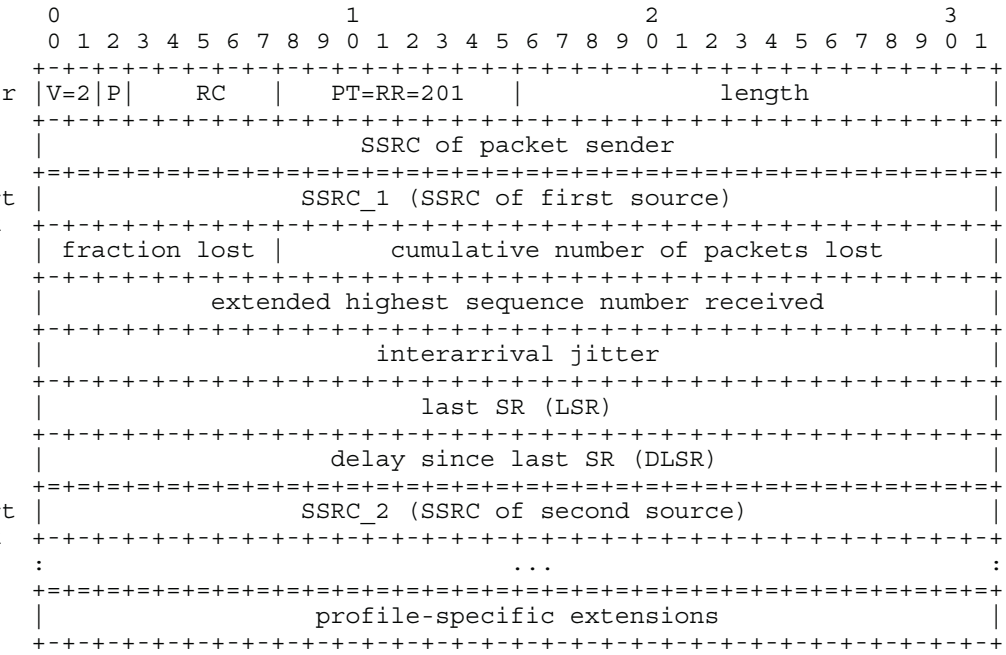

Figure 2: Example for round-trip time computation



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 365
Keysight, EX-1045
Page 365

### 6.4.2 RR: Receiver Report RTCP Packet




The format of the receiver report (RR) packet is the same as that of the SR packet except that the packet type field contains the constant 201 and the five words of sender information are omitted (these are the NTP and RTP timestamps and sender's packet and octet counts). The remaining fields have the same meaning as for the SR packet.

An empty RR packet (RC = 0) MUST be put at the head of a compound RTCP packet when there is no data transmission or reception to report.

### 6.4.3 Extending the Sender and Receiver Reports

A profile SHOULD define profile-specific extensions to the sender report and receiver report if there is additional information that needs to be reported regularly about the sender or receivers. This method SHOULD be used in preference to defining another RTCP packet type because it requires less overhead:

- fewer octets in the packet (no RTCP header or SSRC field);



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 366
Keysight, EX-1045
Page 366

- o simpler and faster parsing because applications running under that profile would be programmed to always expect the extension fields in the directly accessible location after the reception reports.

The extension is a fourth section in the sender- or receiver-report packet which comes at the end after the reception report blocks, if any. If additional sender information is required, then for sender reports it would be included first in the extension section, but for receiver reports it would not be present. If information about receivers is to be included, that data SHOULD be structured as an array of blocks parallel to the existing array of reception report blocks; that is, the number of blocks would be indicated by the RC field.

### 6.4.4 Analyzing Sender and Receiver Reports

It is expected that reception quality feedback will be useful not only for the sender but also for other receivers and third-party monitors. The sender may modify its transmissions based on the feedback; receivers can determine whether problems are local, regional or global; network managers may use profile-independent monitors that receive only the RTCP packets and not the corresponding RTP data packets to evaluate the performance of their networks for multicast distribution.

Cumulative counts are used in both the sender information and receiver report blocks so that differences may be calculated between any two reports to make measurements over both short and long time periods, and to provide resilience against the loss of a report. The difference between the last two reports received can be used to estimate the recent quality of the distribution. The NTP timestamp is included so that rates may be calculated from these differences over the interval between two reports. Since that timestamp is independent of the clock rate for the data encoding, it is possible to implement encoding- and profile-independent quality monitors.

An example calculation is the packet loss rate over the interval between two reception reports. The difference in the cumulative number of packets lost gives the number lost during that interval. The difference in the extended last sequence numbers received gives the number of packets expected during the interval. The ratio of these two is the packet loss fraction over the interval. This ratio should equal the fraction lost field if the two reports are consecutive, but otherwise it may not. The loss rate per second can be obtained by dividing the loss fraction by the difference in NTP timestamps, expressed in seconds. The number of packets received is the number of packets expected minus the number lost. The number of



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 367
Keysight, EX-1045
Page 367

packets expected may also be used to judge the statistical validity of any loss estimates. For example, 1 out of 5 packets lost has a lower significance than 200 out of 1000.

From the sender information, a third-party monitor can calculate the average payload data rate and the average packet rate over an interval without receiving the data. Taking the ratio of the two gives the average payload size. If it can be assumed that packet loss is independent of packet size, then the number of packets received by a particular receiver times the average payload size (or the corresponding packet size) gives the apparent throughput available to that receiver.

In addition to the cumulative counts which allow long-term packet loss measurements using differences between reports, the fraction lost field provides a short-term measurement from a single report. This becomes more important as the size of a session scales up enough that reception state information might not be kept for all receivers or the interval between reports becomes long enough that only one report might have been received from a particular receiver.

The interarrival jitter field provides a second short-term measure of network congestion. Packet loss tracks persistent congestion while the jitter measure tracks transient congestion. The jitter measure may indicate congestion before it leads to packet loss. The interarrival jitter field is only a snapshot of the jitter at the time of a report and is not intended to be taken quantitatively. Rather, it is intended for comparison across a number of reports from one receiver over time or from multiple receivers, e.g., within a single network, at the same time. To allow comparison across receivers, it is important the the jitter be calculated according to the same formula by all receivers.

Because the jitter calculation is based on the RTP timestamp which represents the instant when the first data in the packet was sampled, any variation in the delay between that sampling instant and the time the packet is transmitted will affect the resulting jitter that is calculated. Such a variation in delay would occur for audio packets of varying duration. It will also occur for video encodings because the timestamp is the same for all the packets of one frame but those packets are not all transmitted at the same time. The variation in delay until transmission does reduce the accuracy of the jitter calculation as a measure of the behavior of the network by itself, but it is appropriate to include considering that the receiver buffer must accommodate it. When the jitter calculation is used as a comparative measure, the (constant) component due to variation in delay until transmission subtracts out so that a change in the

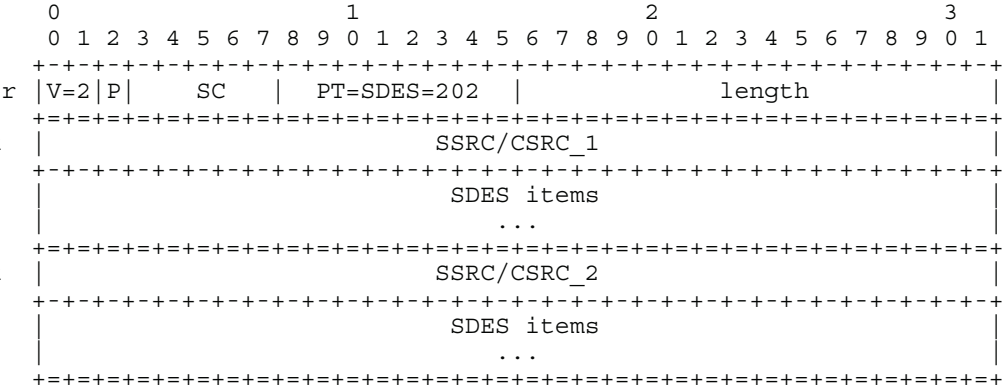


https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 368
Keysight, EX-1045
Page 368

network jitter component can then be observed unless it is relatively small. If the change is small, then it is likely to be inconsequential.

### 6.5 SDES: Source Description RTCP Packet




The SDES packet is a three-level structure composed of a header and zero or more chunks, each of which is composed of items describing the source identified in that chunk. The items are described individually in subsequent sections.

version (V), padding (P), length:
As described for the SR packet (see Section 6.4.1).

packet type (PT): 8 bits
Contains the constant 202 to identify this as an RTCP SDES packet.

source count (SC): 5 bits
The number of SSRC/CSRC chunks contained in this SDES packet. A value of zero is valid but useless.

Each chunk consists of an SSRC/CSRC identifier followed by a list of zero or more items, which carry information about the SSRC/CSRC. Each chunk starts on a 32-bit boundary. Each item consists of an 8-bit type field, an 8-bit octet count describing the length of the text (thus, not including this two-octet header), and the text itself. Note that the text can be no longer than 255 octets, but this is consistent with the need to limit RTCP bandwidth consumption.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 369
Keysight, EX-1045
Page 369

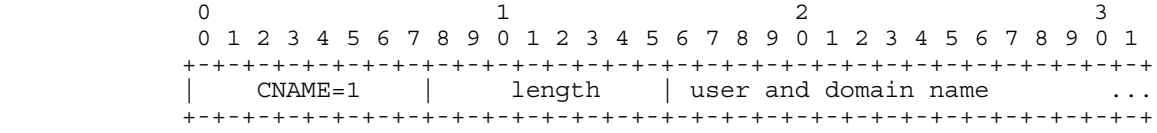
The text is encoded according to the UTF-8 encoding specified in RFC 2279 [5]. US-ASCII is a subset of this encoding and requires no additional encoding. The presence of multi-octet encodings is indicated by setting the most significant bit of a character to a value of one.

Items are contiguous, i.e., items are not individually padded to a 32-bit boundary. Text is not null terminated because some multi-octet encodings include null octets. The list of items in each chunk MUST be terminated by one or more null octets, the first of which is interpreted as an item type of zero to denote the end of the list. No length octet follows the null item type octet, but additional null octets MUST be included if needed to pad until the next 32-bit boundary. Note that this padding is separate from that indicated by the P bit in the RTCP header. A chunk with zero items (four null octets) is valid but useless.

End systems send one SDES packet containing their own source identifier (the same as the SSRC in the fixed RTP header). A mixer sends one SDES packet containing a chunk for each contributing source from which it is receiving SDES information, or multiple complete SDES packets in the format above if there are more than 31 such sources (see Section 7).

The SDES items currently defined are described in the next sections. Only the CNAME item is mandatory. Some items shown here may be useful only for particular profiles, but the item types are all assigned from one common space to promote shared use and to simplify profile-independent applications. Additional items may be defined in a profile by registering the type numbers with IANA as described in Section 15.

### 6.5.1 CNAME: Canonical End-Point Identifier SDES Item




The CNAME identifier has the following properties:

- Because the randomly allocated SSRC identifier may change if a conflict is discovered or if a program is restarted, the CNAME item MUST be included to provide the binding from the SSRC identifier to an identifier for the source (sender or receiver) that remains constant.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 370
Keysight, EX-1045
Page 370

- Like the SSRC identifier, the CNAME identifier SHOULD also be unique among all participants within one RTP session.

- To provide a binding across multiple media tools used by one participant in a set of related RTP sessions, the CNAME SHOULD be fixed for that participant.

- To facilitate third-party monitoring, the CNAME SHOULD be suitable for either a program or a person to locate the source.

Therefore, the CNAME SHOULD be derived algorithmically and not entered manually, when possible. To meet these requirements, the following format SHOULD be used unless a profile specifies an alternate syntax or semantics. The CNAME item SHOULD have the format "user@host", or "host" if a user name is not available as on single-user systems. For both formats, "host" is either the fully qualified domain name of the host from which the real-time data originates, formatted according to the rules specified in RFC 1034 [6], RFC 1035 [7] and Section 2.1 of RFC 1123 [8]; or the standard ASCII representation of the host's numeric address on the interface used for the RTP communication. For example, the standard ASCII representation of an IP Version 4 address is "dotted decimal", also known as dotted quad, and for IP Version 6, addresses are textually represented as groups of hexadecimal digits separated by colons (with variations as detailed in RFC 3513 [23]). Other address types are expected to have ASCII representations that are mutually unique. The fully qualified domain name is more convenient for a human observer and may avoid the need to send a NAME item in addition, but it may be difficult or impossible to obtain reliably in some operating environments. Applications that may be run in such environments SHOULD use the ASCII representation of the address instead.

Examples are "doe@sleepy.example.com", "doe@192.0.2.89" or "doe@2201:056D::112E:144A:1E24" for a multi-user system. On a system with no user name, examples would be "sleepy.example.com", "192.0.2.89" or "2201:056D::112E:144A:1E24".

The user name SHOULD be in a form that a program such as "finger" or "talk" could use, i.e., it typically is the login name rather than the personal name. The host name is not necessarily identical to the one in the participant's electronic mail address.

This syntax will not provide unique identifiers for each source if an application permits a user to generate multiple sources from one host. Such an application would have to rely on the SSRC to further identify the source, or the profile for that application would have to specify additional syntax for the CNAME identifier.



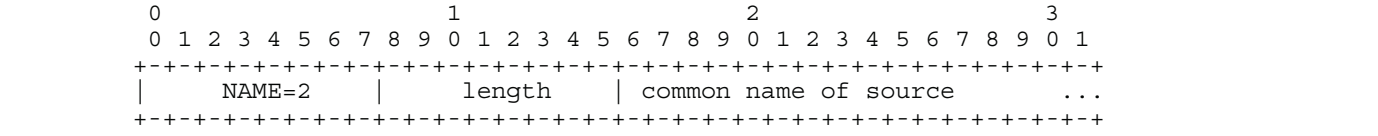
https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 371
Keysight, EX-1045
Page 371

If each application creates its CNAME independently, the resulting CNAMEs may not be identical as would be required to provide a binding across multiple media tools belonging to one participant in a set of related RTP sessions. If cross-media binding is required, it may be necessary for the CNAME of each tool to be externally configured with the same value by a coordination tool.

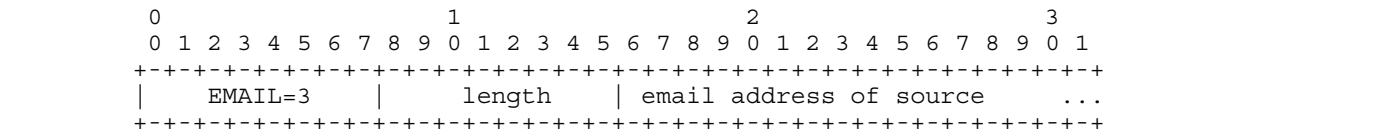
Application writers should be aware that private network address assignments such as the Net-10 assignment proposed in RFC 1918 [24] may create network addresses that are not globally unique. This would lead to non-unique CNAMEs if hosts with private addresses and no direct IP connectivity to the public Internet have their RTP packets forwarded to the public Internet through an RTP-level translator. (See also RFC 1627 [25].) To handle this case, applications MAY provide a means to configure a unique CNAME, but the burden is on the translator to translate CNAMEs from private addresses to public addresses if necessary to keep private addresses from being exposed.

### 6.5.2 NAME: User Name SDES Item




This is the real name used to describe the source, e.g., "John Doe, Bit Recycler". It may be in any form desired by the user. For applications such as conferencing, this form of name may be the most desirable for display in participant lists, and therefore might be sent most frequently of those items other than CNAME. Profiles MAY establish such priorities. The NAME value is expected to remain constant at least for the duration of a session. It SHOULD NOT be relied upon to be unique among all participants in the session.

### 6.5.3 EMAIL: Electronic Mail Address SDES Item



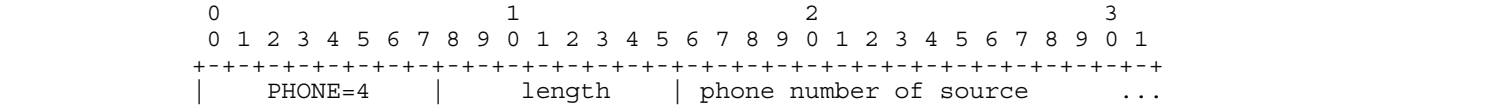

The email address is formatted according to RFC 2822 [9], for example, "John.Doe@example.com". The EMAIL value is expected to remain constant for the duration of a session.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

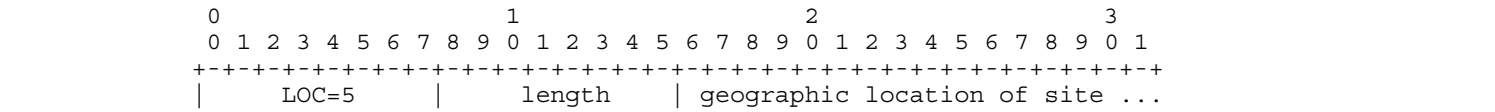
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 372
Keysight, EX-1045
Page 372

### 6.5.4 PHONE: Phone Number SDES Item



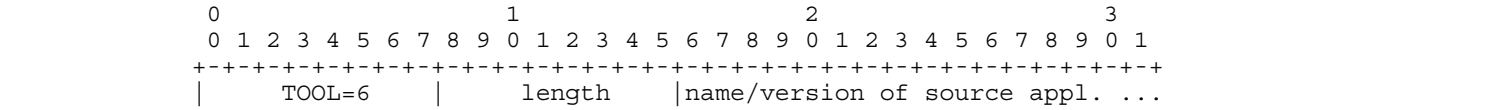

The phone number SHOULD be formatted with the plus sign replacing the international access code. For example, "+1 908 555 1212" for a number in the United States.

### 6.5.5 LOC: Geographic User Location SDES Item




Depending on the application, different degrees of detail are appropriate for this item. For conference applications, a string like "Murray Hill, New Jersey" may be sufficient, while, for an active badge system, strings like "Room 2A244, AT&T BL MH" might be appropriate. The degree of detail is left to the implementation and/or user, but format and content MAY be prescribed by a profile. The LOC value is expected to remain constant for the duration of a session, except for mobile hosts.

### 6.5.6 TOOL: Application or Tool Name SDES Item



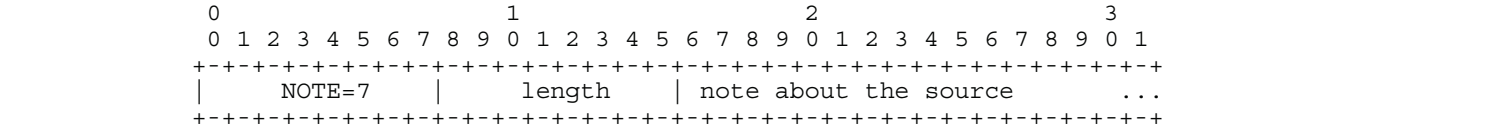

A string giving the name and possibly version of the application generating the stream, e.g., "videotool 1.2". This information may be useful for debugging purposes and is similar to the Mailer or Mail-System-Version SMTP headers. The TOOL value is expected to remain constant for the duration of the session.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 373
Keysight, EX-1045
Page 373

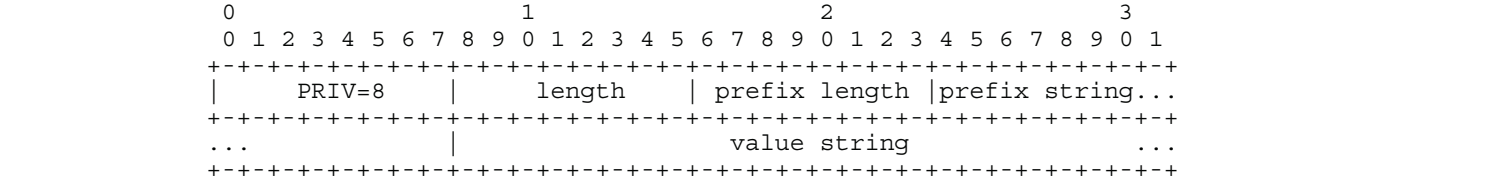
### 6.5.7 NOTE: Notice/Status SDES Item




The following semantics are suggested for this item, but these or other semantics MAY be explicitly defined by a profile. The NOTE item is intended for transient messages describing the current state of the source, e.g., "on the phone, can't talk". Or, during a seminar, this item might be used to convey the title of the talk. It should be used only to carry exceptional information and SHOULD NOT be included routinely by all participants because this would slow down the rate at which reception reports and CNAME are sent, thus impairing the performance of the protocol. In particular, it SHOULD NOT be included as an item in a user's configuration file nor automatically generated as in a quote-of-the-day.

Since the NOTE item may be important to display while it is active, the rate at which other non-CNAME items such as NAME are transmitted might be reduced so that the NOTE item can take that part of the RTCP bandwidth. When the transient message becomes inactive, the NOTE item SHOULD continue to be transmitted a few times at the same repetition rate but with a string of length zero to signal the receivers. However, receivers SHOULD also consider the NOTE item inactive if it is not received for a small multiple of the repetition rate, or perhaps 20-30 RTCP intervals.

### 6.5.8 PRIV: Private Extensions SDES Item




This item is used to define experimental or application-specific SDES extensions. The item contains a prefix consisting of a length-string pair, followed by the value string filling the remainder of the item and carrying the desired information. The prefix length field is 8 bits long. The prefix string is a name chosen by the person defining the PRIV item to be unique with respect to other PRIV items this application might receive. The application creator might choose to use the application name plus an additional subtype identification if



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

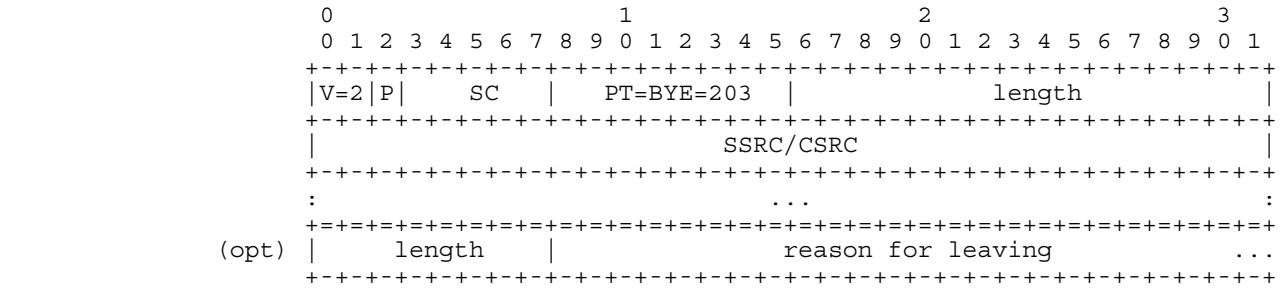
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 374
Keysight, EX-1045
Page 374

needed. Alternatively, it is RECOMMENDED that others choose a name based on the entity they represent, then coordinate the use of the name within that entity.

Note that the prefix consumes some space within the item's total length of 255 octets, so the prefix should be kept as short as possible. This facility and the constrained RTCP bandwidth SHOULD NOT be overloaded; it is not intended to satisfy all the control communication requirements of all applications.

SDES PRIV prefixes will not be registered by IANA. If some form of the PRIV item proves to be of general utility, it SHOULD instead be assigned a regular SDES item type registered with IANA so that no prefix is required. This simplifies use and increases transmission efficiency.

### 6.6 BYE: Goodbye RTCP Packet




The BYE packet indicates that one or more sources are no longer active.

version (V), padding (P), length:
As described for the SR packet (see Section 6.4.1).

packet type (PT): 8 bits
Contains the constant 203 to identify this as an RTCP BYE packet.

source count (SC): 5 bits
The number of SSRC/CSRC identifiers included in this BYE packet. A count value of zero is valid, but useless.

The rules for when a BYE packet should be sent are specified in Sections 6.3.7 and 8.2.

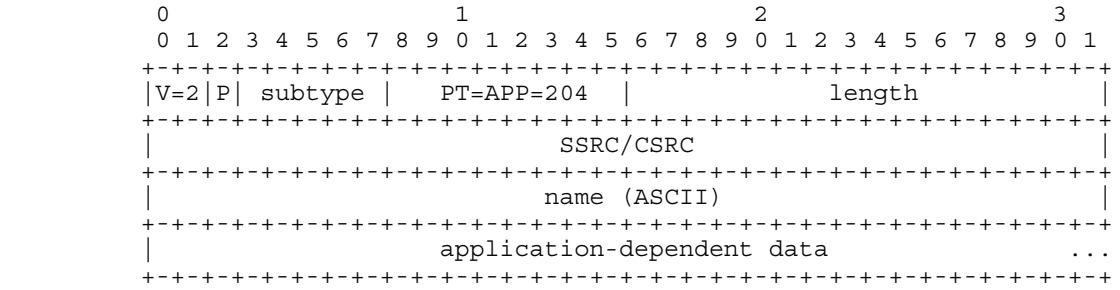


https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 375
Keysight, EX-1045
Page 375

If a BYE packet is received by a mixer, the mixer SHOULD forward the BYE packet with the SSRC/CSRC identifier(s) unchanged. If a mixer shuts down, it SHOULD send a BYE packet listing all contributing sources it handles, as well as its own SSRC identifier. Optionally, the BYE packet MAY include an 8-bit octet count followed by that many octets of text indicating the reason for leaving, e.g., "camera malfunction" or "RTP loop detected". The string has the same encoding as that described for SDES. If the string fills the packet to the next 32-bit boundary, the string is not null terminated. If not, the BYE packet MUST be padded with null octets to the next 32-bit boundary. This padding is separate from that indicated by the P bit in the RTCP header.

### 6.7 APP: Application-Defined RTCP Packet




The APP packet is intended for experimental use as new applications and new features are developed, without requiring packet type value registration. APP packets with unrecognized names SHOULD be ignored. After testing and if wider use is justified, it is RECOMMENDED that each APP packet be redefined without the subtype and name fields and registered with IANA using an RTCP packet type.

version (V), padding (P), length:
As described for the SR packet (see Section 6.4.1).

subtype: 5 bits
May be used as a subtype to allow a set of APP packets to be defined under one unique name, or for any application-dependent data.

packet type (PT): 8 bits
Contains the constant 204 to identify this as an RTCP APP packet.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 376
Keysight, EX-1045
Page 376

name: 4 octets
A name chosen by the person defining the set of APP packets to be unique with respect to other APP packets this application might receive. The application creator might choose to use the application name, and then coordinate the allocation of subtype values to others who want to define new packet types for the application. Alternatively, it is RECOMMENDED that others choose a name based on the entity they represent, then coordinate the use of the name within that entity. The name is interpreted as a sequence of four ASCII characters, with uppercase and lowercase characters treated as distinct.

application-dependent data: variable length
Application-dependent data may or may not appear in an APP packet. It is interpreted by the application and not RTP itself. It MUST be a multiple of 32 bits long.

## 7. RTP Translators and Mixers

In addition to end systems, RTP supports the notion of "translators" and "mixers", which could be considered as "intermediate systems" at the RTP level. Although this support adds some complexity to the protocol, the need for these functions has been clearly established by experiments with multicast audio and video applications in the Internet. Example uses of translators and mixers given in Section 2.3 stem from the presence of firewalls and low bandwidth connections, both of which are likely to remain.

### 7.1 General Description

An RTP translator/mixer connects two or more transport-level "clouds". Typically, each cloud is defined by a common network and transport protocol (e.g., IP/UDP) plus a multicast address and transport level destination port or a pair of unicast addresses and ports. (Network-level protocol translators, such as IP version 4 to IP version 6, may be present within a cloud invisibly to RTP.) One system may serve as a translator or mixer for a number of RTP sessions, but each is considered a logically separate entity.

In order to avoid creating a loop when a translator or mixer is installed, the following rules MUST be observed:

- Each of the clouds connected by translators and mixers participating in one RTP session either MUST be distinct from all the others in at least one of these parameters (protocol, address, port), or MUST be isolated at the network level from the others.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 377
Keysight, EX-1045
Page 377

- o A derivative of the first rule is that there MUST NOT be multiple translators or mixers connected in parallel unless by some arrangement they partition the set of sources to be forwarded.

Similarly, all RTP end systems that can communicate through one or more RTP translators or mixers share the same SSRC space, that is, the SSRC identifiers MUST be unique among all these end systems. Section 8.2 describes the collision resolution algorithm by which SSRC identifiers are kept unique and loops are detected.

There may be many varieties of translators and mixers designed for different purposes and applications. Some examples are to add or remove encryption, change the encoding of the data or the underlying protocols, or replicate between a multicast address and one or more unicast addresses. The distinction between translators and mixers is that a translator passes through the data streams from different sources separately, whereas a mixer combines them to form one new stream:

Translator: Forwards RTP packets with their SSRC identifier intact; this makes it possible for receivers to identify individual sources even though packets from all the sources pass through the same translator and carry the translator's network source address. Some kinds of translators will pass through the data untouched, but others MAY change the encoding of the data and thus the RTP data payload type and timestamp. If multiple data packets are re-encoded into one, or vice versa, a translator MUST assign new sequence numbers to the outgoing packets. Losses in the incoming packet stream may induce corresponding gaps in the outgoing sequence numbers. Receivers cannot detect the presence of a translator unless they know by some other means what payload type or transport address was used by the original source.

Mixer: Receives streams of RTP data packets from one or more sources, possibly changes the data format, combines the streams in some manner and then forwards the combined stream. Since the timing among multiple input sources will not generally be synchronized, the mixer will make timing adjustments among the streams and generate its own timing for the combined stream, so it is the synchronization source. Thus, all data packets forwarded by a mixer MUST be marked with the mixer's own SSRC identifier. In order to preserve the identity of the original sources contributing to the mixed packet, the mixer SHOULD insert their SSRC identifiers into the CSRC identifier list following the fixed RTP header of the packet. A mixer that is also itself a contributing source for some packet SHOULD explicitly include its own SSRC identifier in the CSRC list for that packet.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 378
Keysight, EX-1045
Page 378

For some applications, it MAY be acceptable for a mixer not to identify sources in the CSRC list. However, this introduces the danger that loops involving those sources could not be detected.

The advantage of a mixer over a translator for applications like audio is that the output bandwidth is limited to that of one source even when multiple sources are active on the input side. This may be important for low-bandwidth links. The disadvantage is that receivers on the output side don't have any control over which sources are passed through or muted, unless some mechanism is implemented for remote control of the mixer. The regeneration of synchronization information by mixers also means that receivers can't do inter-media synchronization of the original streams. A multi-media mixer could do it.

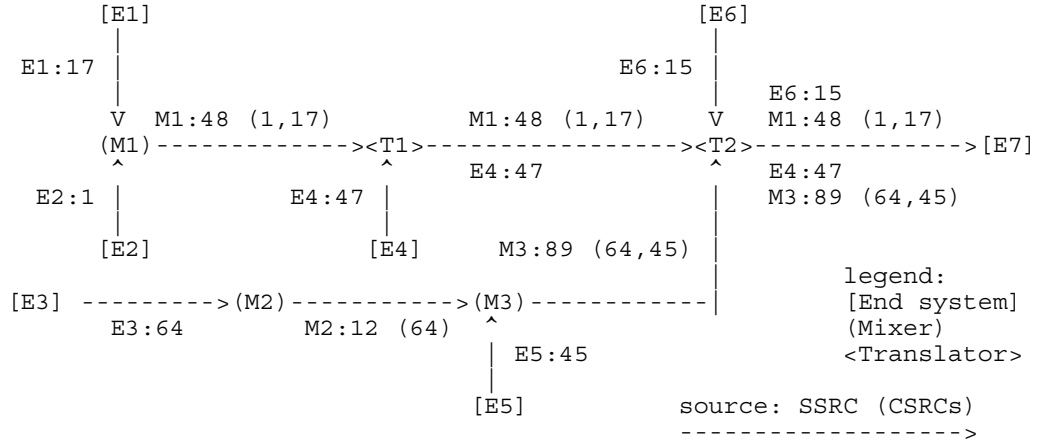



Figure 3: Sample RTP network with end systems, mixers and translators

A collection of mixers and translators is shown in Fig. 3 to illustrate their effect on SSRC and CSRC identifiers. In the figure, end systems are shown as rectangles (named E), translators as triangles (named T) and mixers as ovals (named M). The notation "M1: 48(1,17)" designates a packet originating a mixer M1, identified by M1's (random) SSRC value of 48 and two CSRC identifiers, 1 and 17, copied from the SSRC identifiers of packets from E1 and E2.

### 7.2 RTCP Processing in Translators

In addition to forwarding data packets, perhaps modified, translators and mixers MUST also process RTCP packets. In many cases, they will take apart the compound RTCP packets received from end systems to



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 379
Keysight, EX-1045
Page 379

aggregate SDES information and to modify the SR or RR packets. Retransmission of this information may be triggered by the packet arrival or by the RTCP interval timer of the translator or mixer itself.

A translator that does not modify the data packets, for example one that just replicates between a multicast address and a unicast address, MAY simply forward RTCP packets unmodified as well. A translator that transforms the payload in some way MUST make corresponding transformations in the SR and RR information so that it still reflects the characteristics of the data and the reception quality. These translators MUST NOT simply forward RTCP packets. In general, a translator SHOULD NOT aggregate SR and RR packets from different sources into one packet since that would reduce the accuracy of the propagation delay measurements based on the LSR and DLSR fields.

SR sender information: A translator does not generate its own sender information, but forwards the SR packets received from one cloud to the others. The SSRC is left intact but the sender information MUST be modified if required by the translation. If a translator changes the data encoding, it MUST change the "sender's byte count" field. If it also combines several data packets into one output packet, it MUST change the "sender's packet count" field. If it changes the timestamp frequency, it MUST change the "RTP timestamp" field in the SR packet.

SR/RR reception report blocks: A translator forwards reception reports received from one cloud to the others. Note that these flow in the direction opposite to the data. The SSRC is left intact. If a translator combines several data packets into one output packet, and therefore changes the sequence numbers, it MUST make the inverse manipulation for the packet loss fields and the "extended last sequence number" field. This may be complex. In the extreme case, there may be no meaningful way to translate the reception reports, so the translator MAY pass on no reception report at all or a synthetic report based on its own reception. The general rule is to do what makes sense for a particular translation.

A translator does not require an SSRC identifier of its own, but MAY choose to allocate one for the purpose of sending reports about what it has received. These would be sent to all the connected clouds, each corresponding to the translation of the data stream as sent to that cloud, since reception reports are normally multicast to all participants.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 380
Keysight, EX-1045
Page 380

SDES: Translators typically forward without change the SDES information they receive from one cloud to the others, but MAY, for example, decide to filter non-CNAME SDES information if bandwidth is limited. The CNAMEs MUST be forwarded to allow SSRC identifier collision detection to work. A translator that generates its own RR packets MUST send SDES CNAME information about itself to the same clouds that it sends those RR packets.

BYE: Translators forward BYE packets unchanged. A translator that is about to cease forwarding packets SHOULD send a BYE packet to each connected cloud containing all the SSRC identifiers that were previously being forwarded to that cloud, including the translator's own SSRC identifier if it sent reports of its own.

APP: Translators forward APP packets unchanged.

### 7.3 RTCP Processing in Mixers

Since a mixer generates a new data stream of its own, it does not pass through SR or RR packets at all and instead generates new information for both sides.

SR sender information: A mixer does not pass through sender information from the sources it mixes because the characteristics of the source streams are lost in the mix. As a synchronization source, the mixer SHOULD generate its own SR packets with sender information about the mixed data stream and send them in the same direction as the mixed stream.

SR/RR reception report blocks: A mixer generates its own reception reports for sources in each cloud and sends them out only to the same cloud. It MUST NOT send these reception reports to the other clouds and MUST NOT forward reception reports from one cloud to the others because the sources would not be SSRCs there (only CSRCs).

SDES: Mixers typically forward without change the SDES information they receive from one cloud to the others, but MAY, for example, decide to filter non-CNAME SDES information if bandwidth is limited. The CNAMEs MUST be forwarded to allow SSRC identifier collision detection to work. (An identifier in a CSRC list generated by a mixer might collide with an SSRC identifier generated by an end system.) A mixer MUST send SDES CNAME information about itself to the same clouds that it sends SR or RR packets.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 381
Keysight, EX-1045
Page 381

Since mixers do not forward SR or RR packets, they will typically be extracting SDES packets from a compound RTCP packet. To minimize overhead, chunks from the SDES packets MAY be aggregated into a single SDES packet which is then stacked on an SR or RR packet originating from the mixer. A mixer which aggregates SDES packets will use more RTCP bandwidth than an individual source because the compound packets will be longer, but that is appropriate since the mixer represents multiple sources. Similarly, a mixer which passes through SDES packets as they are received will be transmitting RTCP packets at higher than the single source rate, but again that is correct since the packets come from multiple sources. The RTCP packet rate may be different on each side of the mixer.

A mixer that does not insert CSRC identifiers MAY also refrain from forwarding SDES CNAMEs. In this case, the SSRC identifier spaces in the two clouds are independent. As mentioned earlier, this mode of operation creates a danger that loops can't be detected.

BYE: Mixers MUST forward BYE packets. A mixer that is about to cease forwarding packets SHOULD send a BYE packet to each connected cloud containing all the SSRC identifiers that were previously being forwarded to that cloud, including the mixer's own SSRC identifier if it sent reports of its own.

APP: The treatment of APP packets by mixers is application-specific.

### 7.4 Cascaded Mixers

An RTP session may involve a collection of mixers and translators as shown in Fig. 3. If two mixers are cascaded, such as M2 and M3 in the figure, packets received by a mixer may already have been mixed and may include a CSRC list with multiple identifiers. The second mixer SHOULD build the CSRC list for the outgoing packet using the CSRC identifiers from already-mixed input packets and the SSRC identifiers from unmixed input packets. This is shown in the output arc from mixer M3 labeled M3:89(64,45) in the figure. As in the case of mixers that are not cascaded, if the resulting CSRC list has more than 15 identifiers, the remainder cannot be included.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 382
Keysight, EX-1045
Page 382

## 8. SSRC Identifier Allocation and Use

The SSRC identifier carried in the RTP header and in various fields of RTCP packets is a random 32-bit number that is required to be globally unique within an RTP session. It is crucial that the number be chosen with care in order that participants on the same network or starting at the same time are not likely to choose the same number.

It is not sufficient to use the local network address (such as an IPv4 address) for the identifier because the address may not be unique. Since RTP translators and mixers enable interoperation among multiple networks with different address spaces, the allocation patterns for addresses within two spaces might result in a much higher rate of collision than would occur with random allocation.

Multiple sources running on one host would also conflict.

It is also not sufficient to obtain an SSRC identifier simply by calling random() without carefully initializing the state. An example of how to generate a random identifier is presented in Appendix A.6.

### 8.1 Probability of Collision

Since the identifiers are chosen randomly, it is possible that two or more sources will choose the same number. Collision occurs with the highest probability when all sources are started simultaneously, for example when triggered automatically by some session management event. If N is the number of sources and L the length of the identifier (here, 32 bits), the probability that two sources independently pick the same value can be approximated for large N [26] as 1 - exp(-N**2 / 2**(L+1)). For N=1000, the probability is roughly 10**-4.

The typical collision probability is much lower than the worst-case above. When one new source joins an RTP session in which all the other sources already have unique identifiers, the probability of collision is just the fraction of numbers used out of the space. Again, if N is the number of sources and L the length of the identifier, the probability of collision is N / 2**L. For N=1000, the probability is roughly 2*10**-7.

The probability of collision is further reduced by the opportunity for a new source to receive packets from other participants before sending its first packet (either data or control). If the new source keeps track of the other participants (by SSRC identifier), then



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 383
Keysight, EX-1045
Page 383

before transmitting its first packet the new source can verify that its identifier does not conflict with any that have been received, or else choose again.

### 8.2 Collision Resolution and Loop Detection

Although the probability of SSRC identifier collision is low, all RTP implementations MUST be prepared to detect collisions and take the appropriate actions to resolve them. If a source discovers at any time that another source is using the same SSRC identifier as its own, it MUST send an RTCP BYE packet for the old identifier and choose another random one. (As explained below, this step is taken only once in case of a loop.) If a receiver discovers that two other sources are colliding, it MAY keep the packets from one and discard the packets from the other when this can be detected by different source transport addresses or CNAMEs. The two sources are expected to resolve the collision so that the situation doesn't last.

Because the random SSRC identifiers are kept globally unique for each RTP session, they can also be used to detect loops that may be introduced by mixers or translators. A loop causes duplication of data and control information, either unmodified or possibly mixed, as in the following examples:

- A translator may incorrectly forward a packet to the same multicast group from which it has received the packet, either directly or through a chain of translators. In that case, the same packet appears several times, originating from different network sources.

- Two translators incorrectly set up in parallel, i.e., with the same multicast groups on both sides, would both forward packets from one multicast group to the other. Unidirectional translators would produce two copies; bidirectional translators would form a loop.

- A mixer can close a loop by sending to the same transport destination upon which it receives packets, either directly or through another mixer or translator. In this case a source might show up both as an SSRC on a data packet and a CSRC in a mixed data packet.

A source may discover that its own packets are being looped, or that packets from another source are being looped (a third-party loop). Both loops and collisions in the random selection of a source identifier result in packets arriving with the same SSRC identifier but a different source transport address, which may be that of the end system originating the packet or an intermediate system.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 384
Keysight, EX-1045
Page 384

Therefore, if a source changes its source transport address, it MAY also choose a new SSRC identifier to avoid being interpreted as a looped source. (This is not MUST because in some applications of RTP sources may be expected to change addresses during a session.) Note that if a translator restarts and consequently changes the source transport address (e.g., changes the UDP source port number) on which it forwards packets, then all those packets will appear to receivers to be looped because the SSRC identifiers are applied by the original source and will not change. This problem can be avoided by keeping the source transport address fixed across restarts, but in any case will be resolved after a timeout at the receivers.

Loops or collisions occurring on the far side of a translator or mixer cannot be detected using the source transport address if all copies of the packets go through the translator or mixer, however, collisions may still be detected when chunks from two RTCP SDES packets contain the same SSRC identifier but different CNAMEs.

To detect and resolve these conflicts, an RTP implementation MUST include an algorithm similar to the one described below, though the implementation MAY choose a different policy for which packets from colliding third-party sources are kept. The algorithm described below ignores packets from a new source or loop that collide with an established source. It resolves collisions with the participant's own SSRC identifier by sending an RTCP BYE for the old identifier and choosing a new one. However, when the collision was induced by a loop of the participant's own packets, the algorithm will choose a new identifier only once and thereafter ignore packets from the looping source transport address. This is required to avoid a flood of BYE packets.

This algorithm requires keeping a table indexed by the source identifier and containing the source transport addresses from the first RTP packet and first RTCP packet received with that identifier, along with other state for that source. Two source transport addresses are required since, for example, the UDP source port numbers may be different on RTP and RTCP packets. However, it may be assumed that the network address is the same in both source transport addresses.

Each SSRC or CSRC identifier received in an RTP or RTCP packet is looked up in the source identifier table in order to process that data or control information. The source transport address from the packet is compared to the corresponding source transport address in the table to detect a loop or collision if they don't match. For control packets, each element with its own SSRC identifier, for example an SDES chunk, requires a separate lookup. (The SSRC identifier in a reception report block is an exception because it



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 385
Keysight, EX-1045
Page 385

identifies a source heard by the reporter, and that SSRC identifier is unrelated to the source transport address of the RTCP packet sent by the reporter.) If the SSRC or CSRC is not found, a new entry is created. These table entries are removed when an RTCP BYE packet is received with the corresponding SSRC identifier and validated by a matching source transport address, or after no packets have arrived for a relatively long time (see Section 6.2.1).

Note that if two sources on the same host are transmitting with the same source identifier at the time a receiver begins operation, it would be possible that the first RTP packet received came from one of the sources while the first RTCP packet received came from the other. This would cause the wrong RTCP information to be associated with the RTP data, but this situation should be sufficiently rare and harmless that it may be disregarded.

In order to track loops of the participant's own data packets, the implementation MUST also keep a separate list of source transport addresses (not identifiers) that have been found to be conflicting. As in the source identifier table, two source transport addresses MUST be kept to separately track conflicting RTP and RTCP packets. Note that the conflicting address list should be short, usually empty. Each element in this list stores the source addresses plus the time when the most recent conflicting packet was received. An element MAY be removed from the list when no conflicting packet has arrived from that source for a time on the order of 10 RTCP report intervals (see Section 6.2).

For the algorithm as shown, it is assumed that the participant's own source identifier and state are included in the source identifier table. The algorithm could be restructured to first make a separate comparison against the participant's own source identifier.

```
if (SSRC or CSRC identifier is not found in the source
    identifier table) {
    create a new entry storing the data or control source
        transport address, the SSRC or CSRC and other state;
}

/* Identifier is found in the table */

else if (table entry was created on receipt of a control packet
         and this is the first data packet or vice versa) {
    store the source transport address from this packet;
}
else if (source transport address from the packet does not match
         the one saved in the table entry for this identifier) {
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 386
Keysight, EX-1045
Page 386

```
            /* An identifier collision or a loop is indicated */

            if (source identifier is not the participant's own) {
                /* OPTIONAL error counter step */
                if (source identifier is from an RTCP SDES chunk
                    containing a CNAME item that differs from the CNAME
                    in the table entry) {
                    count a third-party collision;
                } else {
                    count a third-party loop;
                }
                abort processing of data packet or control element;
                /* MAY choose a different policy to keep new source */
            }

            /* A collision or loop of the participant's own packets */

            else if (source transport address is found in the list of
                     conflicting data or control source transport
                     addresses) {
                /* OPTIONAL error counter step */
                if (source identifier is not from an RTCP SDES chunk
                    containing a CNAME item or CNAME is the
                    participant's own) {
                    count occurrence of own traffic looped;
                }
                mark current time in conflicting address list entry;
                abort processing of data packet or control element;
            }

            /* New collision, change SSRC identifier */

            else {
                log occurrence of a collision;
                create a new entry in the conflicting data or control
                    source transport address list and mark current time;
                send an RTCP BYE packet with the old SSRC identifier;
                choose a new SSRC identifier;
                create a new entry in the source identifier table with
                    the old SSRC plus the source transport address from
                    the data or control packet being processed;
            }
        }
```

In this algorithm, packets from a newly conflicting source address will be ignored and packets from the original source address will be kept. If no packets arrive from the original source for an extended period, the table entry will be timed out and the new source will be



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 387
Keysight, EX-1045
Page 387

able to take over. This might occur if the original source detects the collision and moves to a new source identifier, but in the usual case an RTCP BYE packet will be received from the original source to delete the state without having to wait for a timeout.

If the original source address was received through a mixer (i.e., learned as a CSRC) and later the same source is received directly, the receiver may be well advised to switch to the new source address unless other sources in the mix would be lost. Furthermore, for applications such as telephony in which some sources such as mobile entities may change addresses during the course of an RTP session, the RTP implementation SHOULD modify the collision detection algorithm to accept packets from the new source transport address. To guard against flip-flopping between addresses if a genuine collision does occur, the algorithm SHOULD include some means to detect this case and avoid switching.

When a new SSRC identifier is chosen due to a collision, the candidate identifier SHOULD first be looked up in the source identifier table to see if it was already in use by some other source. If so, another candidate MUST be generated and the process repeated.

A loop of data packets to a multicast destination can cause severe network flooding. All mixers and translators MUST implement a loop detection algorithm like the one here so that they can break loops. This should limit the excess traffic to no more than one duplicate copy of the original traffic, which may allow the session to continue so that the cause of the loop can be found and fixed. However, in extreme cases where a mixer or translator does not properly break the loop and high traffic levels result, it may be necessary for end systems to cease transmitting data or control packets entirely. This decision may depend upon the application. An error condition SHOULD be indicated as appropriate. Transmission MAY be attempted again periodically after a long, random time (on the order of minutes).

### 8.3 Use with Layered Encodings

For layered encodings transmitted on separate RTP sessions (see Section 2.4), a single SSRC identifier space SHOULD be used across the sessions of all layers and the core (base) layer SHOULD be used for SSRC identifier allocation and collision resolution. When a source discovers that it has collided, it transmits an RTCP BYE packet on only the base layer but changes the SSRC identifier to the new value in all layers.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 388
Keysight, EX-1045
Page 388

## 9. Security

Lower layer protocols may eventually provide all the security services that may be desired for applications of RTP, including authentication, integrity, and confidentiality. These services have been specified for IP in [27]. Since the initial audio and video applications using RTP needed a confidentiality service before such services were available for the IP layer, the confidentiality service described in the next section was defined for use with RTP and RTCP. That description is included here to codify existing practice. New applications of RTP MAY implement this RTP-specific confidentiality service for backward compatibility, and/or they MAY implement alternative security services. The overhead on the RTP protocol for this confidentiality service is low, so the penalty will be minimal if this service is obsoleted by other services in the future.

Alternatively, other services, other implementations of services and other algorithms may be defined for RTP in the future. In particular, an RTP profile called Secure Real-time Transport Protocol (SRTP) [28] is being developed to provide confidentiality of the RTP payload while leaving the RTP header in the clear so that link-level header compression algorithms can still operate. It is expected that SRTP will be the correct choice for many applications. SRTP is based on the Advanced Encryption Standard (AES) and provides stronger security than the service described here. No claim is made that the methods presented here are appropriate for a particular security need. A profile may specify which services and algorithms should be offered by applications, and may provide guidance as to their appropriate use.

Key distribution and certificates are outside the scope of this document.

### 9.1 Confidentiality

Confidentiality means that only the intended receiver(s) can decode the received packets; for others, the packet contains no useful information. Confidentiality of the content is achieved by encryption.

When it is desired to encrypt RTP or RTCP according to the method specified in this section, all the octets that will be encapsulated for transmission in a single lower-layer packet are encrypted as a unit. For RTCP, a 32-bit random number redrawn for each unit MUST be prepended to the unit before encryption. For RTP, no prefix is prepended; instead, the sequence number and timestamp fields are initialized with random offsets. This is considered to be a weak

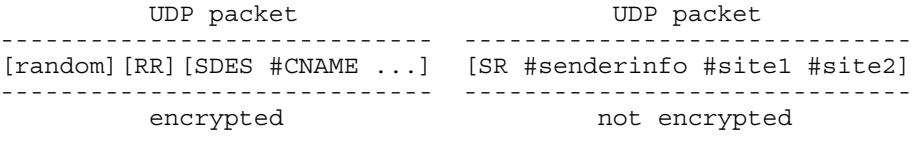


https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 389
Keysight, EX-1045
Page 389

initialization vector (IV) because of poor randomness properties. In addition, if the subsequent field, the SSRC, can be manipulated by an enemy, there is further weakness of the encryption method.

For RTCP, an implementation MAY segregate the individual RTCP packets in a compound RTCP packet into two separate compound RTCP packets, one to be encrypted and one to be sent in the clear. For example, SDES information might be encrypted while reception reports were sent in the clear to accommodate third-party monitors that are not privy to the encryption key. In this example, depicted in Fig. 4, the SDES information MUST be appended to an RR packet with no reports (and the random number) to satisfy the requirement that all compound RTCP packets begin with an SR or RR packet. The SDES CNAME item is required in either the encrypted or unencrypted packet, but not both. The same SDES information SHOULD NOT be carried in both packets as this may compromise the encryption.




#: SSRC identifier

Figure 4: Encrypted and non-encrypted RTCP packets

The presence of encryption and the use of the correct key are confirmed by the receiver through header or payload validity checks. Examples of such validity checks for RTP and RTCP headers are given in Appendices A.1 and A.2.

To be consistent with existing implementations of the initial specification of RTP in RFC 1889, the default encryption algorithm is the Data Encryption Standard (DES) algorithm in cipher block chaining (CBC) mode, as described in Section 1.1 of RFC 1423 [29], except that padding to a multiple of 8 octets is indicated as described for the P bit in Section 5.1. The initialization vector is zero because random values are supplied in the RTP header or by the random prefix for compound RTCP packets. For details on the use of CBC initialization vectors, see [30].

Implementations that support the encryption method specified here SHOULD always support the DES algorithm in CBC mode as the default cipher for this method to maximize interoperability. This method was chosen because it has been demonstrated to be easy and practical to use in experimental audio and video tools in operation on the Internet. However, DES has since been found to be too easily broken.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 390
Keysight, EX-1045
Page 390

It is RECOMMENDED that stronger encryption algorithms such as Triple-DES be used in place of the default algorithm. Furthermore, secure CBC mode requires that the first block of each packet be XORed with a random, independent IV of the same size as the cipher's block size. For RTCP, this is (partially) achieved by prepending each packet with a 32-bit random number, independently chosen for each packet. For RTP, the timestamp and sequence number start from random values, but consecutive packets will not be independently randomized. It should be noted that the randomness in both cases (RTP and RTCP) is limited. High-security applications SHOULD consider other, more conventional, protection means. Other encryption algorithms MAY be specified dynamically for a session by non-RTP means. In particular, the SRTP profile [28] based on AES is being developed to take into account known plaintext and CBC plaintext manipulation concerns, and will be the correct choice in the future.

As an alternative to encryption at the IP level or at the RTP level as described above, profiles MAY define additional payload types for encrypted encodings. Those encodings MUST specify how padding and other aspects of the encryption are to be handled. This method allows encrypting only the data while leaving the headers in the clear for applications where that is desired. It may be particularly useful for hardware devices that will handle both decryption and decoding. It is also valuable for applications where link-level compression of RTP and lower-layer headers is desired and confidentiality of the payload (but not addresses) is sufficient since encryption of the headers precludes compression.

### 9.2 Authentication and Message Integrity

Authentication and message integrity services are not defined at the RTP level since these services would not be directly feasible without a key management infrastructure. It is expected that authentication and integrity services will be provided by lower layer protocols.

## 10. Congestion Control

All transport protocols used on the Internet need to address congestion control in some way [31]. RTP is not an exception, but because the data transported over RTP is often inelastic (generated at a fixed or controlled rate), the means to control congestion in RTP may be quite different from those for other transport protocols such as TCP. In one sense, inelasticity reduces the risk of congestion because the RTP stream will not expand to consume all available bandwidth as a TCP stream can. However, inelasticity also means that the RTP stream cannot arbitrarily reduce its load on the network to eliminate congestion when it occurs.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 391
Keysight, EX-1045
Page 391

Since RTP may be used for a wide variety of applications in many different contexts, there is no single congestion control mechanism that will work for all. Therefore, congestion control SHOULD be defined in each RTP profile as appropriate. For some profiles, it may be sufficient to include an applicability statement restricting the use of that profile to environments where congestion is avoided by engineering. For other profiles, specific methods such as data rate adaptation based on RTCP feedback may be required.

## 11. RTP over Network and Transport Protocols

This section describes issues specific to carrying RTP packets within particular network and transport protocols. The following rules apply unless superseded by protocol-specific definitions outside this specification.

RTP relies on the underlying protocol(s) to provide demultiplexing of RTP data and RTCP control streams. For UDP and similar protocols, RTP SHOULD use an even destination port number and the corresponding RTCP stream SHOULD use the next higher (odd) destination port number. For applications that take a single port number as a parameter and derive the RTP and RTCP port pair from that number, if an odd number is supplied then the application SHOULD replace that number with the next lower (even) number to use as the base of the port pair. For applications in which the RTP and RTCP destination port numbers are specified via explicit, separate parameters (using a signaling protocol or other means), the application MAY disregard the restrictions that the port numbers be even/odd and consecutive although the use of an even/odd port pair is still encouraged. The RTP and RTCP port numbers MUST NOT be the same since RTP relies on the port numbers to demultiplex the RTP data and RTCP control streams.

In a unicast session, both participants need to identify a port pair for receiving RTP and RTCP packets. Both participants MAY use the same port pair. A participant MUST NOT assume that the source port of the incoming RTP or RTCP packet can be used as the destination port for outgoing RTP or RTCP packets. When RTP data packets are being sent in both directions, each participant's RTCP SR packets MUST be sent to the port that the other participant has specified for reception of RTCP. The RTCP SR packets combine sender information for the outgoing data plus reception report information for the incoming data. If a side is not actively sending data (see Section 6.4), an RTCP RR packet is sent instead.

It is RECOMMENDED that layered encoding applications (see Section 2.4) use a set of contiguous port numbers. The port numbers MUST be distinct because of a widespread deficiency in existing operating



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 392
Keysight, EX-1045
Page 392

systems that prevents use of the same port with multiple multicast addresses, and for unicast, there is only one permissible address. Thus for layer n, the data port is P + 2n, and the control port is P + 2n + 1. When IP multicast is used, the addresses MUST also be distinct because multicast routing and group membership are managed on an address granularity. However, allocation of contiguous IP multicast addresses cannot be assumed because some groups may require different scopes and may therefore be allocated from different address ranges.

The previous paragraph conflicts with the SDP specification, RFC 2327 [15], which says that it is illegal for both multiple addresses and multiple ports to be specified in the same session description because the association of addresses with ports could be ambiguous. It is intended that this restriction will be relaxed in a revision of RFC 2327 to allow an equal number of addresses and ports to be specified with a one-to-one mapping implied.

RTP data packets contain no length field or other delineation, therefore RTP relies on the underlying protocol(s) to provide a length indication. The maximum length of RTP packets is limited only by the underlying protocols.

If RTP packets are to be carried in an underlying protocol that provides the abstraction of a continuous octet stream rather than messages (packets), an encapsulation of the RTP packets MUST be defined to provide a framing mechanism. Framing is also needed if the underlying protocol may contain padding so that the extent of the RTP payload cannot be determined. The framing mechanism is not defined here.

A profile MAY specify a framing method to be used even when RTP is carried in protocols that do provide framing in order to allow carrying several RTP packets in one lower-layer protocol data unit, such as a UDP packet. Carrying several RTP packets in one network or transport packet reduces header overhead and may simplify synchronization between different streams.

## 12. Summary of Protocol Constants

This section contains a summary listing of the constants defined in this specification.

The RTP payload type (PT) constants are defined in profiles rather than this document. However, the octet of the RTP header which contains the marker bit(s) and payload type MUST avoid the reserved values 200 and 201 (decimal) to distinguish RTP packets from the RTCP SR and RR packet types for the header validation procedure described



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 393
Keysight, EX-1045
Page 393

in Appendix A.1. For the standard definition of one marker bit and a 7-bit payload type field as shown in this specification, this restriction means that payload types 72 and 73 are reserved.

### 12.1 RTCP Packet Types

| abbrev. | name | value |
|---|---|---|
| SR | sender report | 200 |
| RR | receiver report | 201 |
| SDES | source description | 202 |
| BYE | goodbye | 203 |
| APP | application-defined | 204 |

These type values were chosen in the range 200-204 for improved header validity checking of RTCP packets compared to RTP packets or other unrelated packets. When the RTCP packet type field is compared to the corresponding octet of the RTP header, this range corresponds to the marker bit being 1 (which it usually is not in data packets) and to the high bit of the standard payload type field being 1 (since the static payload types are typically defined in the low half). This range was also chosen to be some distance numerically from 0 and 255 since all-zeros and all-ones are common data patterns.

Since all compound RTCP packets MUST begin with SR or RR, these codes were chosen as an even/odd pair to allow the RTCP validity check to test the maximum number of bits with mask and value.

Additional RTCP packet types may be registered through IANA (see Section 15).

### 12.2 SDES Types

| abbrev. | name | value |
|---|---|---|
| END | end of SDES list | 0 |
| CNAME | canonical name | 1 |
| NAME | user name | 2 |
| EMAIL | user's electronic mail address | 3 |
| PHONE | user's phone number | 4 |
| LOC | geographic user location | 5 |
| TOOL | name of application or tool | 6 |
| NOTE | notice about the source | 7 |
| PRIV | private extensions | 8 |

Additional SDES types may be registered through IANA (see Section 15).



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 394
Keysight, EX-1045
Page 394

## 13. RTP Profiles and Payload Format Specifications

A complete specification of RTP for a particular application will require one or more companion documents of two types described here: profiles, and payload format specifications.

RTP may be used for a variety of applications with somewhat differing requirements. The flexibility to adapt to those requirements is provided by allowing multiple choices in the main protocol specification, then selecting the appropriate choices or defining extensions for a particular environment and class of applications in a separate profile document. Typically an application will operate under only one profile in a particular RTP session, so there is no explicit indication within the RTP protocol itself as to which profile is in use. A profile for audio and video applications may be found in the companion RFC 3551. Profiles are typically titled "RTP Profile for ...".

The second type of companion document is a payload format specification, which defines how a particular kind of payload data, such as H.261 encoded video, should be carried in RTP. These documents are typically titled "RTP Payload Format for XYZ Audio/Video Encoding". Payload formats may be useful under multiple profiles and may therefore be defined independently of any particular profile. The profile documents are then responsible for assigning a default mapping of that format to a payload type value if needed.

Within this specification, the following items have been identified for possible definition within a profile, but this list is not meant to be exhaustive:

RTP data header: The octet in the RTP data header that contains the marker bit and payload type field MAY be redefined by a profile to suit different requirements, for example with more or fewer marker bits (Section 5.3, p. 18).

Payload types: Assuming that a payload type field is included, the profile will usually define a set of payload formats (e.g., media encodings) and a default static mapping of those formats to payload type values. Some of the payload formats may be defined by reference to separate payload format specifications. For each payload type defined, the profile MUST specify the RTP timestamp clock rate to be used (Section 5.1, p. 14).

RTP data header additions: Additional fields MAY be appended to the fixed RTP data header if some additional functionality is required across the profile's class of applications independent of payload type (Section 5.3, p. 18).



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 395
Keysight, EX-1045
Page 395

RTP data header extensions: The contents of the first 16 bits of the RTP data header extension structure MUST be defined if use of that mechanism is to be allowed under the profile for implementation-specific extensions (Section 5.3.1, p. 18).

RTCP packet types: New application-class-specific RTCP packet types MAY be defined and registered with IANA.

RTCP report interval: A profile SHOULD specify that the values suggested in Section 6.2 for the constants employed in the calculation of the RTCP report interval will be used. Those are the RTCP fraction of session bandwidth, the minimum report interval, and the bandwidth split between senders and receivers. A profile MAY specify alternate values if they have been demonstrated to work in a scalable manner.

SR/RR extension: An extension section MAY be defined for the RTCP SR and RR packets if there is additional information that should be reported regularly about the sender or receivers (Section 6.4.3, p. 42 and 43).

SDES use: The profile MAY specify the relative priorities for RTCP SDES items to be transmitted or excluded entirely (Section 6.3.9); an alternate syntax or semantics for the CNAME item (Section 6.5.1); the format of the LOC item (Section 6.5.5); the semantics and use of the NOTE item (Section 6.5.7); or new SDES item types to be registered with IANA.

Security: A profile MAY specify which security services and algorithms should be offered by applications, and MAY provide guidance as to their appropriate use (Section 9, p. 65).

String-to-key mapping: A profile MAY specify how a user-provided password or pass phrase is mapped into an encryption key.

Congestion: A profile SHOULD specify the congestion control behavior appropriate for that profile.

Underlying protocol: Use of a particular underlying network or transport layer protocol to carry RTP packets MAY be required.

Transport mapping: A mapping of RTP and RTCP to transport-level addresses, e.g., UDP ports, other than the standard mapping defined in Section 11, p. 68 may be specified.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 396
Keysight, EX-1045
Page 396

Encapsulation: An encapsulation of RTP packets may be defined to allow multiple RTP data packets to be carried in one lower-layer packet or to provide framing over underlying protocols that do not already do so (Section 11, p. 69).

It is not expected that a new profile will be required for every application. Within one application class, it would be better to extend an existing profile rather than make a new one in order to facilitate interoperation among the applications since each will typically run under only one profile. Simple extensions such as the definition of additional payload type values or RTCP packet types may be accomplished by registering them through IANA and publishing their descriptions in an addendum to the profile or in a payload format specification.

## 14. Security Considerations

RTP suffers from the same security liabilities as the underlying protocols. For example, an impostor can fake source or destination network addresses, or change the header or payload. Within RTCP, the CNAME and NAME information may be used to impersonate another participant. In addition, RTP may be sent via IP multicast, which provides no direct means for a sender to know all the receivers of the data sent and therefore no measure of privacy. Rightly or not, users may be more sensitive to privacy concerns with audio and video communication than they have been with more traditional forms of network communication [33]. Therefore, the use of security mechanisms with RTP is important. These mechanisms are discussed in Section 9.

RTP-level translators or mixers may be used to allow RTP traffic to reach hosts behind firewalls. Appropriate firewall security principles and practices, which are beyond the scope of this document, should be followed in the design and installation of these devices and in the admission of RTP applications for use behind the firewall.

## 15. IANA Considerations

Additional RTCP packet types and SDES item types may be registered through the Internet Assigned Numbers Authority (IANA). Since these number spaces are small, allowing unconstrained registration of new values would not be prudent. To facilitate review of requests and to promote shared use of new types among multiple applications, requests for registration of new values must be documented in an RFC or other permanent and readily available reference such as the product of another cooperative standards body (e.g., ITU-T). Other requests may also be accepted, under the advice of a "designated expert."



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 397
Keysight, EX-1045
Page 397

(Contact the IANA for the contact information of the current expert.)

RTP profile specifications SHOULD register with IANA a name for the profile in the form "RTP/xxx", where xxx is a short abbreviation of the profile title. These names are for use by higher-level control protocols, such as the Session Description Protocol (SDP), RFC 2327 [15], to refer to transport methods.

## 16. Intellectual Property Rights Statement

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

## 17. Acknowledgments

This memorandum is based on discussions within the IETF Audio/Video Transport working group chaired by Stephen Casner and Colin Perkins. The current protocol has its origins in the Network Voice Protocol and the Packet Video Protocol (Danny Cohen and Randy Cole) and the protocol implemented by the vat application (Van Jacobson and Steve McCanne). Christian Huitema provided ideas for the random identifier generator. Extensive analysis and simulation of the timer reconsideration algorithm was done by Jonathan Rosenberg. The additions for layered encodings were specified by Michael Speer and Steve McCanne.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 398
Keysight, EX-1045
Page 398

## Appendix A - Algorithms

We provide examples of C code for aspects of RTP sender and receiver algorithms. There may be other implementation methods that are faster in particular operating environments or have other advantages. These implementation notes are for informational purposes only and are meant to clarify the RTP specification.

The following definitions are used for all examples; for clarity and brevity, the structure definitions are only valid for 32-bit big-endian (most significant octet first) architectures. Bit fields are assumed to be packed tightly in big-endian bit order, with no additional padding. Modifications would be required to construct a portable implementation.

```
/*
 * rtp.h  --  RTP header file
 */
#include <sys/types.h>

/*
 * The type definitions below are valid for 32-bit architectures and
 * may have to be adjusted for 16- or 64-bit architectures.
 */
typedef unsigned char  u_int8;
typedef unsigned short u_int16;
typedef unsigned int   u_int32;
typedef          short int16;

/*
 * Current protocol version.
 */
#define RTP_VERSION    2

#define RTP_SEQ_MOD (1<<16)
#define RTP_MAX_SDES 255      /* maximum text length for SDES */

typedef enum {
    RTCP_SR   = 200,
    RTCP_RR   = 201,
    RTCP_SDES = 202,
    RTCP_BYE  = 203,
    RTCP_APP  = 204
} rtcp_type_t;

typedef enum {
    RTCP_SDES_END   = 0,
    RTCP_SDES_CNAME = 1,
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 399
Keysight, EX-1045
Page 399

```
        RTCP_SDES_NAME  = 2,
        RTCP_SDES_EMAIL = 3,
        RTCP_SDES_PHONE = 4,
        RTCP_SDES_LOC   = 5,
        RTCP_SDES_TOOL  = 6,
        RTCP_SDES_NOTE  = 7,
        RTCP_SDES_PRIV  = 8
    } rtcp_sdes_type_t;

    /*
     * RTP data header
     */
    typedef struct {
        unsigned int version:2;   /* protocol version */
        unsigned int p:1;         /* padding flag */
        unsigned int x:1;         /* header extension flag */
        unsigned int cc:4;        /* CSRC count */
        unsigned int m:1;         /* marker bit */
        unsigned int pt:7;        /* payload type */
        unsigned int seq:16;      /* sequence number */
        u_int32 ts;               /* timestamp */
        u_int32 ssrc;             /* synchronization source */
        u_int32 csrc[1];          /* optional CSRC list */
    } rtp_hdr_t;

    /*
     * RTCP common header word
     */
    typedef struct {
        unsigned int version:2;   /* protocol version */
        unsigned int p:1;         /* padding flag */
        unsigned int count:5;     /* varies by packet type */
        unsigned int pt:8;        /* RTCP packet type */
        u_int16 length;           /* pkt len in words, w/o this word */
    } rtcp_common_t;

    /*
     * Big-endian mask for version, padding bit and packet type pair
     */
    #define RTCP_VALID_MASK (0xc000 | 0x2000 | 0xfe)
    #define RTCP_VALID_VALUE ((RTP_VERSION << 14) | RTCP_SR)

    /*
     * Reception report block
     */
    typedef struct {
        u_int32 ssrc;             /* data source being reported */
        unsigned int fraction:8;  /* fraction lost since last SR/RR */
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 400
Keysight, EX-1045
Page 400

```
        int lost:24;              /* cumul. no. pkts lost (signed!) */
        u_int32 last_seq;         /* extended last seq. no. received */
        u_int32 jitter;           /* interarrival jitter */
        u_int32 lsr;              /* last SR packet from this source */
        u_int32 dlsr;             /* delay since last SR packet */
    } rtcp_rr_t;

    /*
     * SDES item
     */
    typedef struct {
        u_int8 type;              /* type of item (rtcp_sdes_type_t) */
        u_int8 length;            /* length of item (in octets) */
        char data[1];             /* text, not null-terminated */
    } rtcp_sdes_item_t;

    /*
     * One RTCP packet
     */
    typedef struct {
        rtcp_common_t common;     /* common header */
        union {
            /* sender report (SR) */
            struct {
                u_int32 ssrc;     /* sender generating this report */
                u_int32 ntp_sec;  /* NTP timestamp */
                u_int32 ntp_frac;
                u_int32 rtp_ts;   /* RTP timestamp */
                u_int32 psent;    /* packets sent */
                u_int32 osent;    /* octets sent */
                rtcp_rr_t rr[1];  /* variable-length list */
            } sr;

            /* reception report (RR) */
            struct {
                u_int32 ssrc;     /* receiver generating this report */
                rtcp_rr_t rr[1];  /* variable-length list */
            } rr;

            /* source description (SDES) */
            struct rtcp_sdes {
                u_int32 src;      /* first SSRC/CSRC */
                rtcp_sdes_item_t item[1]; /* list of SDES items */
            } sdes;

            /* BYE */
            struct {
                u_int32 src[1];   /* list of sources */
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 401
Keysight, EX-1045
Page 401

```
            /* can't express trailing text for reason */
        } bye;
    } r;
} rtcp_t;

typedef struct rtcp_sdes rtcp_sdes_t;

/*
 * Per-source state information
 */
typedef struct {
    u_int16 max_seq;        /* highest seq. number seen */
    u_int32 cycles;         /* shifted count of seq. number cycles */
    u_int32 base_seq;       /* base seq number */
    u_int32 bad_seq;        /* last 'bad' seq number + 1 */
    u_int32 probation;      /* sequ. packets till source is valid */
    u_int32 received;       /* packets received */
    u_int32 expected_prior; /* packet expected at last interval */
    u_int32 received_prior; /* packet received at last interval */
    u_int32 transit;        /* relative trans time for prev pkt */
    u_int32 jitter;         /* estimated jitter */
    /* ... */
} source;
```

## A.1 RTP Data Header Validity Checks

An RTP receiver should check the validity of the RTP header on incoming packets since they might be encrypted or might be from a different application that happens to be misaddressed. Similarly, if encryption according to the method described in Section 9 is enabled, the header validity check is needed to verify that incoming packets have been correctly decrypted, although a failure of the header validity check (e.g., unknown payload type) may not necessarily indicate decryption failure.

Only weak validity checks are possible on an RTP data packet from a source that has not been heard before:

- RTP version field must equal 2.
- The payload type must be known, and in particular it must not be equal to SR or RR.
- If the P bit is set, then the last octet of the packet must contain a valid octet count, in particular, less than the total packet length minus the header size.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 402
Keysight, EX-1045
Page 402

- The X bit must be zero if the profile does not specify that the header extension mechanism may be used. Otherwise, the extension length field must be less than the total packet size minus the fixed header length and padding.

- The length of the packet must be consistent with CC and payload type (if payloads have a known length).

The last three checks are somewhat complex and not always possible, leaving only the first two which total just a few bits. If the SSRC identifier in the packet is one that has been received before, then the packet is probably valid and checking if the sequence number is in the expected range provides further validation. If the SSRC identifier has not been seen before, then data packets carrying that identifier may be considered invalid until a small number of them arrive with consecutive sequence numbers. Those invalid packets MAY be discarded or they MAY be stored and delivered once validation has been achieved if the resulting delay is acceptable.

The routine update_seq shown below ensures that a source is declared valid only after MIN_SEQUENTIAL packets have been received in sequence. It also validates the sequence number seq of a newly received packet and updates the sequence state for the packet's source in the structure to which s points.

When a new source is heard for the first time, that is, its SSRC identifier is not in the table (see Section 8.2), and the per-source state is allocated for it, s->probation is set to the number of sequential packets required before declaring a source valid (parameter MIN_SEQUENTIAL) and other variables are initialized:

```
init_seq(s, seq);
s->max_seq = seq - 1;
s->probation = MIN_SEQUENTIAL;
```

A non-zero s->probation marks the source as not yet valid so the state may be discarded after a short timeout rather than a long one, as discussed in Section 6.2.1.

After a source is considered valid, the sequence number is considered valid if it is no more than MAX_DROPOUT ahead of s->max_seq nor more than MAX_MISORDER behind. If the new sequence number is ahead of max_seq modulo the RTP sequence number range (16 bits), but is smaller than max_seq, it has wrapped around and the (shifted) count of sequence number cycles is incremented. A value of one is returned to indicate a valid sequence number.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 403
Keysight, EX-1045
Page 403

Otherwise, the value zero is returned to indicate that the validation failed, and the bad sequence number plus 1 is stored. If the next packet received carries the next higher sequence number, it is considered the valid start of a new packet sequence presumably caused by an extended dropout or a source restart. Since multiple complete sequence number cycles may have been missed, the packet loss statistics are reset.

Typical values for the parameters are shown, based on a maximum misordering time of 2 seconds at 50 packets/second and a maximum dropout of 1 minute. The dropout parameter MAX_DROPOUT should be a small fraction of the 16-bit sequence number space to give a reasonable probability that new sequence numbers after a restart will not fall in the acceptable range for sequence numbers from before the restart.

```
void init_seq(source *s, u_int16 seq)
{
    s->base_seq = seq;
    s->max_seq = seq;
    s->bad_seq = RTP_SEQ_MOD + 1;   /* so seq == bad_seq is false */
    s->cycles = 0;
    s->received = 0;
    s->received_prior = 0;
    s->expected_prior = 0;
    /* other initialization */
}

int update_seq(source *s, u_int16 seq)
{
    u_int16 udelta = seq - s->max_seq;
    const int MAX_DROPOUT = 3000;
    const int MAX_MISORDER = 100;
    const int MIN_SEQUENTIAL = 2;

    /*
     * Source is not valid until MIN_SEQUENTIAL packets with
     * sequential sequence numbers have been received.
     */
    if (s->probation) {
        /* packet is in sequence */
        if (seq == s->max_seq + 1) {
            s->probation--;
            s->max_seq = seq;
            if (s->probation == 0) {
                init_seq(s, seq);
                s->received++;
                return 1;
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 404
Keysight, EX-1045
Page 404

```
                }
            } else {
                s->probation = MIN_SEQUENTIAL - 1;
                s->max_seq = seq;
            }
            return 0;
        } else if (udelta < MAX_DROPOUT) {
            /* in order, with permissible gap */
            if (seq < s->max_seq) {
                /*
                 * Sequence number wrapped - count another 64K cycle.
                 */
                s->cycles += RTP_SEQ_MOD;
            }
            s->max_seq = seq;
        } else if (udelta <= RTP_SEQ_MOD - MAX_MISORDER) {
            /* the sequence number made a very large jump */
            if (seq == s->bad_seq) {
                /*
                 * Two sequential packets -- assume that the other side
                 * restarted without telling us so just re-sync
                 * (i.e., pretend this was the first packet).
                 */
                init_seq(s, seq);
            }
            else {
                s->bad_seq = (seq + 1) & (RTP_SEQ_MOD-1);
                return 0;
            }
        } else {
            /* duplicate or reordered packet */
        }
        s->received++;
        return 1;
    }
```

The validity check can be made stronger requiring more than two packets in sequence. The disadvantages are that a larger number of initial packets will be discarded (or delayed in a queue) and that high packet loss rates could prevent validation. However, because the RTCP header validation is relatively strong, if an RTCP packet is received from a source before the data packets, the count could be adjusted so that only two packets are required in sequence. If initial data loss for a few seconds can be tolerated, an application MAY choose to discard all data packets from a source until a valid RTCP packet has been received from that source.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 405
Keysight, EX-1045
Page 405

Depending on the application and encoding, algorithms may exploit additional knowledge about the payload format for further validation. For payload types where the timestamp increment is the same for all packets, the timestamp values can be predicted from the previous packet received from the same source using the sequence number difference (assuming no change in payload type).

A strong "fast-path" check is possible since with high probability the first four octets in the header of a newly received RTP data packet will be just the same as that of the previous packet from the same SSRC except that the sequence number will have increased by one. Similarly, a single-entry cache may be used for faster SSRC lookups in applications where data is typically received from one source at a time.

### A.2 RTCP Header Validity Checks

The following checks should be applied to RTCP packets.

- RTP version field must equal 2.
- The payload type field of the first RTCP packet in a compound packet must be equal to SR or RR.
- The padding bit (P) should be zero for the first packet of a compound RTCP packet because padding should only be applied, if it is needed, to the last packet.
- The length fields of the individual RTCP packets must add up to the overall length of the compound RTCP packet as received. This is a fairly strong check.

The code fragment below performs all of these checks. The packet type is not checked for subsequent packets since unknown packet types may be present and should be ignored.

```
u_int32 len;        /* length of compound RTCP packet in words */
rtcp_t *r;          /* RTCP header */
rtcp_t *end;        /* end of compound RTCP packet */

if ((*(u_int16 *)r & RTCP_VALID_MASK) != RTCP_VALID_VALUE) {
    /* something wrong with packet format */
}
end = (rtcp_t *)((u_int32 *)r + len);

do r = (rtcp_t *)((u_int32 *)r + r->common.length + 1);
while (r < end && r->common.version == 2);
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 406
Keysight, EX-1045
Page 406

```
      if (r != end) {
          /* something wrong with packet format */
      }
```

### A.3 Determining Number of Packets Expected and Lost

In order to compute packet loss rates, the number of RTP packets expected and actually received from each source needs to be known, using per-source state information defined in struct source referenced via pointer s in the code below. The number of packets received is simply the count of packets as they arrive, including any late or duplicate packets. The number of packets expected can be computed by the receiver as the difference between the highest sequence number received (s->max_seq) and the first sequence number received (s->base_seq). Since the sequence number is only 16 bits and will wrap around, it is necessary to extend the highest sequence number with the (shifted) count of sequence number wraparounds (s->cycles). Both the received packet count and the count of cycles are maintained the RTP header validity check routine in Appendix A.1.

```
   extended_max = s->cycles + s->max_seq;
   expected = extended_max - s->base_seq + 1;
```

The number of packets lost is defined to be the number of packets expected less the number of packets actually received:

```
   lost = expected - s->received;
```

Since this signed number is carried in 24 bits, it should be clamped at 0x7fffff for positive loss or 0x800000 for negative loss rather than wrapping around.

The fraction of packets lost during the last reporting interval (since the previous SR or RR packet was sent) is calculated from differences in the expected and received packet counts across the interval, where expected_prior and received_prior are the values saved when the previous reception report was generated:

```
   expected_interval = expected - s->expected_prior;
   s->expected_prior = expected;
   received_interval = s->received - s->received_prior;
   s->received_prior = s->received;
   lost_interval = expected_interval - received_interval;
   if (expected_interval == 0 || lost_interval <= 0) fraction = 0;
   else fraction = (lost_interval << 8) / expected_interval;
```

The resulting fraction is an 8-bit fixed point number with the binary point at the left edge.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 407
Keysight, EX-1045
Page 407

## A.4 Generating RTCP SDES Packets

This function builds one SDES chunk into buffer b composed of argc items supplied in arrays type, value and length. It returns a pointer to the next available location within b.

```
char *rtp_write_sdes(char *b, u_int32 src, int argc,
                     rtcp_sdes_type_t type[], char *value[],
                     int length[])
{
    rtcp_sdes_t *s = (rtcp_sdes_t *)b;
    rtcp_sdes_item_t *rsp;
    int i;
    int len;
    int pad;

    /* SSRC header */
    s->src = src;
    rsp = &s->item[0];

    /* SDES items */
    for (i = 0; i < argc; i++) {
        rsp->type = type[i];
        len = length[i];
        if (len > RTP_MAX_SDES) {
            /* invalid length, may want to take other action */
            len = RTP_MAX_SDES;
        }
        rsp->length = len;
        memcpy(rsp->data, value[i], len);
        rsp = (rtcp_sdes_item_t *)&rsp->data[len];
    }

    /* terminate with end marker and pad to next 4-octet boundary */
    len = ((char *) rsp) - b;
    pad = 4 - (len & 0x3);
    b = (char *) rsp;
    while (pad--) *b++ = RTCP_SDES_END;

    return b;
}
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 408
Keysight, EX-1045
Page 408

## A.5 Parsing RTCP SDES Packets

This function parses an SDES packet, calling functions find_member() to find a pointer to the information for a session member given the SSRC identifier and member_sdes() to store the new SDES information for that member. This function expects a pointer to the header of the RTCP packet.

```
void rtp_read_sdes(rtcp_t *r)
{
    int count = r->common.count;
    rtcp_sdes_t *sd = &r->r.sdes;
    rtcp_sdes_item_t *rsp, *rspn;
    rtcp_sdes_item_t *end = (rtcp_sdes_item_t *)
                            ((u_int32 *)r + r->common.length + 1);
    source *s;

    while (--count >= 0) {
        rsp = &sd->item[0];
        if (rsp >= end) break;
        s = find_member(sd->src);

        for (; rsp->type; rsp = rspn ) {
            rspn = (rtcp_sdes_item_t *)((char*)rsp+rsp->length+2);
            if (rspn >= end) {
                rsp = rspn;
                break;
            }
            member_sdes(s, rsp->type, rsp->data, rsp->length);
        }
        sd = (rtcp_sdes_t *)
             ((u_int32 *)sd + (((char *)rsp - (char *)sd) >> 2)+1);
    }
    if (count >= 0) {
        /* invalid packet format */
    }
}
```

## A.6 Generating a Random 32-bit Identifier

The following subroutine generates a random 32-bit identifier using the MD5 routines published in RFC 1321 [32]. The system routines may not be present on all operating systems, but they should serve as hints as to what kinds of information may be used. Other system calls that may be appropriate include



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 409
Keysight, EX-1045
Page 409

- getdomainname(),
- getwd(), or
- getrusage().

"Live" video or audio samples are also a good source of random numbers, but care must be taken to avoid using a turned-off microphone or blinded camera as a source [17].

Use of this or a similar routine is recommended to generate the initial seed for the random number generator producing the RTCP period (as shown in Appendix A.7), to generate the initial values for the sequence number and timestamp, and to generate SSRC values. Since this routine is likely to be CPU-intensive, its direct use to generate RTCP periods is inappropriate because predictability is not an issue. Note that this routine produces the same result on repeated calls until the value of the system clock changes unless different values are supplied for the type argument.

```
/*
 * Generate a random 32-bit quantity.
 */
#include <sys/types.h>   /* u_long */
#include <sys/time.h>    /* gettimeofday() */
#include <unistd.h>      /* get..() */
#include <stdio.h>       /* printf() */
#include <time.h>        /* clock() */
#include <sys/utsname.h> /* uname() */
#include "global.h"      /* from RFC 1321 */
#include "md5.h"         /* from RFC 1321 */

#define MD_CTX MD5_CTX
#define MDInit MD5Init
#define MDUpdate MD5Update
#define MDFinal MD5Final

static u_long md_32(char *string, int length)
{
    MD_CTX context;
    union {
        char   c[16];
        u_long x[4];
    } digest;
    u_long r;
    int i;

    MDInit (&context);
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 410
Keysight, EX-1045
Page 410

```
        MDUpdate (&context, string, length);
        MDFinal ((unsigned char *)&digest, &context);
        r = 0;
        for (i = 0; i < 3; i++) {
            r ^= digest.x[i];
        }
        return r;
    }                               /* md_32 */

    /*
     * Return random unsigned 32-bit quantity.  Use 'type' argument if
     * you need to generate several different values in close succession.
     */
    u_int32 random32(int type)
    {
        struct {
            int     type;
            struct  timeval tv;
            clock_t cpu;
            pid_t   pid;
            u_long  hid;
            uid_t   uid;
            gid_t   gid;
            struct  utsname name;
        } s;

        gettimeofday(&s.tv, 0);
        uname(&s.name);
        s.type = type;
        s.cpu  = clock();
        s.pid  = getpid();
        s.hid  = gethostid();
        s.uid  = getuid();
        s.gid  = getgid();
        /* also: system uptime */

        return md_32((char *)&s, sizeof(s));
    }                               /* random32 */
```

## A.7 Computing the RTCP Transmission Interval

The following functions implement the RTCP transmission and reception rules described in Section 6.2. These rules are coded in several functions:

- rtcp_interval() computes the deterministic calculated interval, measured in seconds. The parameters are defined in Section 6.3.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 411
Keysight, EX-1045
Page 411

- OnExpire() is called when the RTCP transmission timer expires.

- OnReceive() is called whenever an RTCP packet is received.

Both OnExpire() and OnReceive() have event e as an argument. This is the next scheduled event for that participant, either an RTCP report or a BYE packet. It is assumed that the following functions are available:

- Schedule(time t, event e) schedules an event e to occur at time t. When time t arrives, the function OnExpire is called with e as an argument.

- Reschedule(time t, event e) reschedules a previously scheduled event e for time t.

- SendRTCPReport(event e) sends an RTCP report.

- SendBYEPacket(event e) sends a BYE packet.

- TypeOfEvent(event e) returns EVENT_BYE if the event being processed is for a BYE packet to be sent, else it returns EVENT_REPORT.

- PacketType(p) returns PACKET_RTCP_REPORT if packet p is an RTCP report (not BYE), PACKET_BYE if its a BYE RTCP packet, and PACKET_RTP if its a regular RTP data packet.

- ReceivedPacketSize() and SentPacketSize() return the size of the referenced packet in octets.

- NewMember(p) returns a 1 if the participant who sent packet p is not currently in the member list, 0 otherwise. Note this function is not sufficient for a complete implementation because each CSRC identifier in an RTP packet and each SSRC in a BYE packet should be processed.

- NewSender(p) returns a 1 if the participant who sent packet p is not currently in the sender sublist of the member list, 0 otherwise.

- AddMember() and RemoveMember() to add and remove participants from the member list.

- AddSender() and RemoveSender() to add and remove participants from the sender sublist of the member list.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 412
Keysight, EX-1045
Page 412

These functions would have to be extended for an implementation that allows the RTCP bandwidth fractions for senders and non-senders to be specified as explicit parameters rather than fixed values of 25% and 75%. The extended implementation of rtcp_interval() would need to avoid division by zero if one of the parameters was zero.

```
double rtcp_interval(int members,
                     int senders,
                     double rtcp_bw,
                     int we_sent,
                     double avg_rtcp_size,
                     int initial)
{
    /*
     * Minimum average time between RTCP packets from this site (in
     * seconds).  This time prevents the reports from `clumping' when
     * sessions are small and the law of large numbers isn't helping
     * to smooth out the traffic.  It also keeps the report interval
     * from becoming ridiculously small during transient outages like
     * a network partition.
     */
    double const RTCP_MIN_TIME = 5.;
    /*
     * Fraction of the RTCP bandwidth to be shared among active
     * senders.  (This fraction was chosen so that in a typical
     * session with one or two active senders, the computed report
     * time would be roughly equal to the minimum report time so that
     * we don't unnecessarily slow down receiver reports.)  The
     * receiver fraction must be 1 - the sender fraction.
     */
    double const RTCP_SENDER_BW_FRACTION = 0.25;
    double const RTCP_RCVR_BW_FRACTION = (1-RTCP_SENDER_BW_FRACTION);
    /*
    /* To compensate for "timer reconsideration" converging to a
     * value below the intended average.
     */
    double const COMPENSATION = 2.71828 - 1.5;

    double t;                   /* interval */
    double rtcp_min_time = RTCP_MIN_TIME;
    int n;                      /* no. of members for computation */

    /*
     * Very first call at application start-up uses half the min
     * delay for quicker notification while still allowing some time
     * before reporting for randomization and to learn about other
     * sources so the report interval will converge to the correct
     * interval more quickly.
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 413
Keysight, EX-1045
Page 413

```
         */
        if (initial) {
            rtcp_min_time /= 2;
        }
        /*
         * Dedicate a fraction of the RTCP bandwidth to senders unless
         * the number of senders is large enough that their share is
         * more than that fraction.
         */
        n = members;
        if (senders <= members * RTCP_SENDER_BW_FRACTION) {
            if (we_sent) {
                rtcp_bw *= RTCP_SENDER_BW_FRACTION;
                n = senders;
            } else {
                rtcp_bw *= RTCP_RCVR_BW_FRACTION;
                n -= senders;
            }
        }

        /*
         * The effective number of sites times the average packet size is
         * the total number of octets sent when each site sends a report.
         * Dividing this by the effective bandwidth gives the time
         * interval over which those packets must be sent in order to
         * meet the bandwidth target, with a minimum enforced.  In that
         * time interval we send one report so this time is also our
         * average time between reports.
         */
        t = avg_rtcp_size * n / rtcp_bw;
        if (t < rtcp_min_time) t = rtcp_min_time;

        /*
         * To avoid traffic bursts from unintended synchronization with
         * other sites, we then pick our actual next report interval as a
         * random number uniformly distributed between 0.5*t and 1.5*t.
         */
        t = t * (drand48() + 0.5);
        t = t / COMPENSATION;
        return t;
    }

    void OnExpire(event e,
                  int    members,
                  int    senders,
                  double rtcp_bw,
                  int    we_sent,
                  double *avg_rtcp_size,
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 414
Keysight, EX-1045
Page 414

```
                int    *initial,
                time_tp   tc,
                time_tp   *tp,
                int    *pmembers)
   {
       /* This function is responsible for deciding whether to send an
        * RTCP report or BYE packet now, or to reschedule transmission.
        * It is also responsible for updating the pmembers, initial, tp,
        * and avg_rtcp_size state variables.  This function should be
        * called upon expiration of the event timer used by Schedule().
        */

       double t;     /* Interval */
       double tn;    /* Next transmit time */

       /* In the case of a BYE, we use "timer reconsideration" to
        * reschedule the transmission of the BYE if necessary */

       if (TypeOfEvent(e) == EVENT_BYE) {
           t = rtcp_interval(members,
                             senders,
                             rtcp_bw,
                             we_sent,
                             *avg_rtcp_size,
                             *initial);
           tn = *tp + t;
           if (tn <= tc) {
               SendBYEPacket(e);
               exit(1);
           } else {
               Schedule(tn, e);
           }

       } else if (TypeOfEvent(e) == EVENT_REPORT) {
           t = rtcp_interval(members,
                             senders,
                             rtcp_bw,
                             we_sent,
                             *avg_rtcp_size,
                             *initial);
           tn = *tp + t;
           if (tn <= tc) {
               SendRTCPReport(e);
               *avg_rtcp_size = (1./16.)*SentPacketSize(e) +
                   (15./16.)*(*avg_rtcp_size);
               *tp = tc;

               /* We must redraw the interval.  Don't reuse the
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 415
Keysight, EX-1045
Page 415

```
                one computed above, since its not actually
                distributed the same, as we are conditioned
                on it being small enough to cause a packet to
                be sent */

             t = rtcp_interval(members,
                               senders,
                               rtcp_bw,
                               we_sent,
                               *avg_rtcp_size,
                               *initial);

             Schedule(t+tc,e);
             *initial = 0;
          } else {
             Schedule(tn, e);
          }
          *pmembers = members;
       }
    }

    void OnReceive(packet p,
                   event e,
                   int *members,
                   int *pmembers,
                   int *senders,
                   double *avg_rtcp_size,
                   double *tp,
                   double tc,
                   double tn)
    {
        /* What we do depends on whether we have left the group, and are
         * waiting to send a BYE (TypeOfEvent(e) == EVENT_BYE) or an RTCP
         * report.  p represents the packet that was just received.  */

        if (PacketType(p) == PACKET_RTCP_REPORT) {
            if (NewMember(p) && (TypeOfEvent(e) == EVENT_REPORT)) {
                AddMember(p);
                *members += 1;
            }
            *avg_rtcp_size = (1./16.)*ReceivedPacketSize(p) +
                (15./16.)*(*avg_rtcp_size);
        } else if (PacketType(p) == PACKET_RTP) {
            if (NewMember(p) && (TypeOfEvent(e) == EVENT_REPORT)) {
                AddMember(p);
                *members += 1;
            }
            if (NewSender(p) && (TypeOfEvent(e) == EVENT_REPORT)) {
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 416
Keysight, EX-1045
Page 416

```
                AddSender(p);
                *senders += 1;
            }
        } else if (PacketType(p) == PACKET_BYE) {
            *avg_rtcp_size = (1./16.)*ReceivedPacketSize(p) +
                (15./16.)*(*avg_rtcp_size);

            if (TypeOfEvent(e) == EVENT_REPORT) {
                if (NewSender(p) == FALSE) {
                    RemoveSender(p);
                    *senders -= 1;
                }

                if (NewMember(p) == FALSE) {
                    RemoveMember(p);
                    *members -= 1;
                }

                if (*members < *pmembers) {
                    tn = tc +
                         (((double) *members)/(*pmembers))*(tn - tc);
                    *tp = tc -
                         (((double) *members)/(*pmembers))*(tc - *tp);

                    /* Reschedule the next report for time tn */

                    Reschedule(tn, e);
                    *pmembers = *members;
                }

            } else if (TypeOfEvent(e) == EVENT_BYE) {
                *members += 1;
            }
        }
    }
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 417
Keysight, EX-1045
Page 417

## A.8 Estimating the Interarrival Jitter

The code fragments below implement the algorithm given in Section 6.4.1 for calculating an estimate of the statistical variance of the RTP data interarrival time to be inserted in the interarrival jitter field of reception reports. The inputs are r->ts, the timestamp from the incoming packet, and arrival, the current time in the same units. Here s points to state for the source; s->transit holds the relative transit time for the previous packet, and s->jitter holds the estimated jitter. The jitter field of the reception report is measured in timestamp units and expressed as an unsigned integer, but the jitter estimate is kept in a floating point. As each data packet arrives, the jitter estimate is updated:

```
int transit = arrival - r->ts;
int d = transit - s->transit;
s->transit = transit;
if (d < 0) d = -d;
s->jitter += (1./16.) * ((double)d - s->jitter);
```

When a reception report block (to which rr points) is generated for this member, the current jitter estimate is returned:

```
rr->jitter = (u_int32) s->jitter;
```

Alternatively, the jitter estimate can be kept as an integer, but scaled to reduce round-off error. The calculation is the same except for the last line:

```
s->jitter += d - ((s->jitter + 8) >> 4);
```

In this case, the estimate is sampled for the reception report as:

```
rr->jitter = s->jitter >> 4;
```



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 418
Keysight, EX-1045
Page 418

## Appendix B - Changes from RFC 1889

Most of this RFC is identical to RFC 1889. There are no changes in the packet formats on the wire, only changes to the rules and algorithms governing how the protocol is used. The biggest change is an enhancement to the scalable timer algorithm for calculating when to send RTCP packets:

- The algorithm for calculating the RTCP transmission interval specified in Sections 6.2 and 6.3 and illustrated in Appendix A.7 is augmented to include "reconsideration" to minimize transmission in excess of the intended rate when many participants join a session simultaneously, and "reverse reconsideration" to reduce the incidence and duration of false participant timeouts when the number of participants drops rapidly. Reverse reconsideration is also used to possibly shorten the delay before sending RTCP SR when transitioning from passive receiver to active sender mode.

- Section 6.3.7 specifies new rules controlling when an RTCP BYE packet should be sent in order to avoid a flood of packets when many participants leave a session simultaneously.

- The requirement to retain state for inactive participants for a period long enough to span typical network partitions was removed from Section 6.2.1. In a session where many participants join for a brief time and fail to send BYE, this requirement would cause a significant overestimate of the number of participants. The reconsideration algorithm added in this revision compensates for the large number of new participants joining simultaneously when a partition heals.

It should be noted that these enhancements only have a significant effect when the number of session participants is large (thousands) and most of the participants join or leave at the same time. This makes testing in a live network difficult. However, the algorithm was subjected to a thorough analysis and simulation to verify its performance. Furthermore, the enhanced algorithm was designed to interoperate with the algorithm in RFC 1889 such that the degree of reduction in excess RTCP bandwidth during a step join is proportional to the fraction of participants that implement the enhanced algorithm. Interoperation of the two algorithms has been verified experimentally on live networks.

Other functional changes were:

- Section 6.2.1 specifies that implementations may store only a sampling of the participants' SSRC identifiers to allow scaling to very large sessions. Algorithms are specified in RFC 2762 [21].



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 419
Keysight, EX-1045
Page 419

- In Section 6.2 it is specified that RTCP sender and non-sender bandwidths may be set as separate parameters of the session rather than a strict percentage of the session bandwidth, and may be set to zero. The requirement that RTCP was mandatory for RTP sessions using IP multicast was relaxed. However, a clarification was also added that turning off RTCP is NOT RECOMMENDED.

- In Sections 6.2, 6.3.1 and Appendix A.7, it is specified that the fraction of participants below which senders get dedicated RTCP bandwidth changes from the fixed 1/4 to a ratio based on the RTCP sender and non-sender bandwidth parameters when those are given. The condition that no bandwidth is dedicated to senders when there are no senders was removed since that is expected to be a transitory state. It also keeps non-senders from using sender RTCP bandwidth when that is not intended.

- Also in Section 6.2 it is specified that the minimum RTCP interval may be scaled to smaller values for high bandwidth sessions, and that the initial RTCP delay may be set to zero for unicast sessions.

- Timing out a participant is to be based on inactivity for a number of RTCP report intervals calculated using the receiver RTCP bandwidth fraction even for active senders.

- Sections 7.2 and 7.3 specify that translators and mixers should send BYE packets for the sources they are no longer forwarding.

- Rule changes for layered encodings are defined in Sections 2.4, 6.3.9, 8.3 and 11. In the last of these, it is noted that the address and port assignment rule conflicts with the SDP specification, RFC 2327 [15], but it is intended that this restriction will be relaxed in a revision of RFC 2327.

- The convention for using even/odd port pairs for RTP and RTCP in Section 11 was clarified to refer to destination ports. The requirement to use an even/odd port pair was removed if the two ports are specified explicitly. For unicast RTP sessions, distinct port pairs may be used for the two ends (Sections 3, 7.1 and 11).

- A new Section 10 was added to explain the requirement for congestion control in applications using RTP.

- In Section 8.2, the requirement that a new SSRC identifier MUST be chosen whenever the source transport address is changed has been relaxed to say that a new SSRC identifier MAY be chosen. Correspondingly, it was clarified that an implementation MAY



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 420
Keysight, EX-1045
Page 420

choose to keep packets from the new source address rather than the existing source address when an SSRC collision occurs between two other participants, and SHOULD do so for applications such as telephony in which some sources such as mobile entities may change addresses during the course of an RTP session.

- An indentation bug in the RFC 1889 printing of the pseudo-code for the collision detection and resolution algorithm in Section 8.2 has been corrected by translating the syntax to pseudo C language, and the algorithm has been modified to remove the restriction that both RTP and RTCP must be sent from the same source port number.

- The description of the padding mechanism for RTCP packets was clarified and it is specified that padding MUST only be applied to the last packet of a compound RTCP packet.

- In Section A.1, initialization of base_seq was corrected to be seq rather than seq - 1, and the text was corrected to say the bad sequence number plus 1 is stored. The initialization of max_seq and other variables for the algorithm was separated from the text to make clear that this initialization must be done in addition to calling the init_seq() function (and a few words lost in RFC 1889 when processing the document from source to output form were restored).

- Clamping of number of packets lost in Section A.3 was corrected to use both positive and negative limits.

- The specification of "relative" NTP timestamp in the RTCP SR section now defines these timestamps to be based on the most common system-specific clock, such as system uptime, rather than on session elapsed time which would not be the same for multiple applications started on the same machine at different times.

Non-functional changes:

- It is specified that a receiver MUST ignore packets with payload types it does not understand.

- In Fig. 2, the floating point NTP timestamp value was corrected, some missing leading zeros were added in a hex number, and the UTC timezone was specified.

- The inconsequence of NTP timestamps wrapping around in the year 2036 is explained.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 421
Keysight, EX-1045
Page 421

- The policy for registration of RTCP packet types and SDES types was clarified in a new Section 15, IANA Considerations. The suggestion that experimenters register the numbers they need and then unregister those which prove to be unneeded has been removed in favor of using APP and PRIV. Registration of profile names was also specified.

- The reference for the UTF-8 character set was changed from an X/Open Preliminary Specification to be RFC 2279.

- The reference for RFC 1597 was updated to RFC 1918 and the reference for RFC 2543 was updated to RFC 3261.

- The last paragraph of the introduction in RFC 1889, which cautioned implementors to limit deployment in the Internet, was removed because it was deemed no longer relevant.

- A non-normative note regarding the use of RTP with Source-Specific Multicast (SSM) was added in Section 6.

- The definition of "RTP session" in Section 3 was expanded to acknowledge that a single session may use multiple destination transport addresses (as was always the case for a translator or mixer) and to explain that the distinguishing feature of an RTP session is that each corresponds to a separate SSRC identifier space. A new definition of "multimedia session" was added to reduce confusion about the word "session".

- The meaning of "sampling instant" was explained in more detail as part of the definition of the timestamp field of the RTP header in Section 5.1.

- Small clarifications of the text have been made in several places, some in response to questions from readers. In particular:

  - In RFC 1889, the first five words of the second sentence of Section 2.2 were lost in processing the document from source to output form, but are now restored.

  - A definition for "RTP media type" was added in Section 3 to allow the explanation of multiplexing RTP sessions in Section 5.2 to be more clear regarding the multiplexing of multiple media. That section also now explains that multiplexing multiple sources of the same medium based on SSRC identifiers may be appropriate and is the norm for multicast sessions.

  - The definition for "non-RTP means" was expanded to include examples of other protocols constituting non-RTP means.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 422
Keysight, EX-1045
Page 422

- The description of the session bandwidth parameter is expanded in Section 6.2, including a clarification that the control traffic bandwidth is in addition to the session bandwidth for the data traffic.

- The effect of varying packet duration on the jitter calculation was explained in Section 6.4.4.

- The method for terminating and padding a sequence of SDES items was clarified in Section 6.5.

- IPv6 address examples were added in the description of SDES CNAME in Section 6.5.1, and "example.com" was used in place of other example domain names.

- The Security section added a formal reference to IPSEC now that it is available, and says that the confidentiality method defined in this specification is primarily to codify existing practice. It is RECOMMENDED that stronger encryption algorithms such as Triple-DES be used in place of the default algorithm, and noted that the SRTP profile based on AES will be the correct choice in the future. A caution about the weakness of the RTP header as an initialization vector was added. It was also noted that payload-only encryption is necessary to allow for header compression.

- The method for partial encryption of RTCP was clarified; in particular, SDES CNAME is carried in only one part when the compound RTCP packet is split.

- It is clarified that only one compound RTCP packet should be sent per reporting interval and that if there are too many active sources for the reports to fit in the MTU, then a subset of the sources should be selected round-robin over multiple intervals.

- A note was added in Appendix A.1 that packets may be saved during RTP header validation and delivered upon success.

- Section 7.3 now explains that a mixer aggregating SDES packets uses more RTCP bandwidth due to longer packets, and a mixer passing through RTCP naturally sends packets at higher than the single source rate, but both behaviors are valid.

- Section 13 clarifies that an RTP application may use multiple profiles but typically only one in a given session.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 423
Keysight, EX-1045
Page 423

- The terms MUST, SHOULD, MAY, etc. are used as defined in RFC 2119.

- The bibliography was divided into normative and informative references.

## References

### Normative References

[1] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", RFC 3551, July 2003.

[2] Bradner, S., "Key Words for Use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[3] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[4] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation and Analysis", RFC 1305, March 1992.

[5] Yergeau, F., "UTF-8, a Transformation Format of ISO 10646", RFC 2279, January 1998.

[6] Mockapetris, P., "Domain Names - Concepts and Facilities", STD 13, RFC 1034, November 1987.

[7] Mockapetris, P., "Domain Names - Implementation and Specification", STD 13, RFC 1035, November 1987.

[8] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989.

[9] Resnick, P., "Internet Message Format", RFC 2822, April 2001.

### Informative References

[10] Clark, D. and D. Tennenhouse, "Architectural Considerations for a New Generation of Protocols," in SIGCOMM Symposium on Communications Architectures and Protocols , (Philadelphia, Pennsylvania), pp. 200--208, IEEE Computer Communications Review, Vol. 20(4), September 1990.

[11] Schulzrinne, H., "Issues in designing a transport protocol for audio and video conferences and other multiparticipant real-time applications." expired Internet Draft, October 1993.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 424
Keysight, EX-1045
Page 424

[12] Comer, D., Internetworking with TCP/IP , vol. 1. Englewood Cliffs, New Jersey: Prentice Hall, 1991.

[13] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[14] International Telecommunication Union, "Visual telephone systems and equipment for local area networks which provide a non-guaranteed quality of service", Recommendation H.323, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, July 2003.

[15] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

[16] Schulzrinne, H., Rao, A. and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

[17] Eastlake 3rd, D., Crocker, S. and J. Schiller, "Randomness Recommendations for Security", RFC 1750, December 1994.

[18] Bolot, J.-C., Turletti, T. and I. Wakeman, "Scalable Feedback Control for Multicast Video Distribution in the Internet", in SIGCOMM Symposium on Communications Architectures and Protocols, (London, England), pp. 58--67, ACM, August 1994.

[19] Busse, I., Deffner, B. and H. Schulzrinne, "Dynamic QoS Control of Multimedia Applications Based on RTP", Computer Communications , vol. 19, pp. 49--58, January 1996.

[20] Floyd, S. and V. Jacobson, "The Synchronization of Periodic Routing Messages", in SIGCOMM Symposium on Communications Architectures and Protocols (D. P. Sidhu, ed.), (San Francisco, California), pp. 33--44, ACM, September 1993. Also in [34].

[21] Rosenberg, J. and H. Schulzrinne, "Sampling of the Group Membership in RTP", RFC 2762, February 2000.

[22] Cadzow, J., Foundations of Digital Signal Processing and Data Analysis New York, New York: Macmillan, 1987.

[23] Hinden, R. and S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture", RFC 3513, April 2003.

[24] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G. and E. Lear, "Address Allocation for Private Internets", RFC 1918, February 1996.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 425
Keysight, EX-1045
Page 425

[25] Lear, E., Fair, E., Crocker, D. and T. Kessler, "Network 10 Considered Harmful (Some Practices Shouldn't be Codified)", RFC 1627, July 1994.

[26] Feller, W., An Introduction to Probability Theory and its Applications, vol. 1. New York, New York: John Wiley and Sons, third ed., 1968.

[27] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[28] Baugher, M., Blom, R., Carrara, E., McGrew, D., Naslund, M., Norrman, K. and D. Oran, "Secure Real-time Transport Protocol", Work in Progress, April 2003.

[29] Balenson, D., "Privacy Enhancement for Internet Electronic Mail: Part III", RFC 1423, February 1993.

[30] Voydock, V. and S. Kent, "Security Mechanisms in High-Level Network Protocols", ACM Computing Surveys, vol. 15, pp. 135-171, June 1983.

[31] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, September 2000.

[32] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

[33] Stubblebine, S., "Security Services for Multimedia Conferencing", in 16th National Computer Security Conference, (Baltimore, Maryland), pp. 391--395, September 1993.

[34] Floyd, S. and V. Jacobson, "The Synchronization of Periodic Routing Messages", IEEE/ACM Transactions on Networking, vol. 2, pp. 122--136, April 1994.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 426
Keysight, EX-1045
Page 426

## Authors' Addresses

Henning Schulzrinne
Department of Computer Science
Columbia University
1214 Amsterdam Avenue
New York, NY 10027
United States

EMail: schulzrinne@cs.columbia.edu

Stephen L. Casner
Packet Design
3400 Hillview Avenue, Building 3
Palo Alto, CA 94304
United States

EMail: casner@acm.org

Ron Frederick
Blue Coat Systems Inc.
650 Almanor Avenue
Sunnyvale, CA 94085
United States

EMail: ronf@bluecoat.com

Van Jacobson
Packet Design
3400 Hillview Avenue, Building 3
Palo Alto, CA 94304
United States

EMail: van@packetdesign.com



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 427
Keysight, EX-1045
Page 427

Full Copyright Statement

Copyright (C) The Internet Society (2003). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.



https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 428
Keysight, EX-1045
Page 428

https://web.archive.org/web/20070206011723/https://tools.ietf.org/html/rfc3550

| | |
|---|---|
| Network Working Group | P. Saint-Andre, Ed. |
| Request for Comments: 3920 | Jabber Software Foundation |
| Category: Standards Track | October 2004 |


RFC 3920 TOC

# Extensible Messaging and Presence Protocol (XMPP): Core

## Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Copyright Notice

Copyright © The Internet Society (2004).

## Abstract

This memo defines the core features of the Extensible Messaging and Presence Protocol (XMPP), a protocol for streaming Extensible Markup Language (XML) elements in order to exchange structured information in close to real time between any two network endpoints. While XMPP provides a generalized, extensible framework for exchanging XML data, it is used mainly for the purpose of building instant messaging and presence applications that meet the requirements of RFC 2779.

---

## Table of Contents

RFC 3920 TOC


1. Introduction
2. Generalized Architecture
3. Addressing Scheme
4. XML Streams
5. Use of TLS
6. Use of SASL
7. Resource Binding
8. Server Dialback
9. XML Stanzas
10. Server Rules for Handling XML Stanzas
11. XML Usage within XMPP
12. Core Compliance Requirements
13. Internationalization Considerations
14. Security Considerations
15. IANA Considerations
16. References
    - 16.1. Normative References
    - 16.2. Informative References

Appendix A. Nodeprep
Appendix B. Resourceprep
Appendix C. XML Schemas


https://web.archive.org/web/20061010092431/https://xmpp.org/rfcs/rfc3920.html

**Appendix D. Differences Between Core Jabber Protocols and XMPP**
**Appendix E. Contributors**
**Appendix F. Acknowledgements**
**§ Author's Address**
**§ Intellectual Property and Copyright Statements**

## 1. Introduction

TOC

### 1.1. Overview

TOC

The Extensible Messaging and Presence Protocol (XMPP) is an open Extensible Markup Language **XML** [XML] protocol for near-real-time messaging, presence, and request-response services. The basic syntax and semantics were developed originally within the Jabber open-source community, mainly in 1999. In 2002, the XMPP WG was chartered with developing an adaptation of the Jabber protocol that would be suitable as an IETF instant messaging (IM) and presence technology. As a result of work by the XMPP WG, the current memo defines the core features of XMPP 1.0; the extensions required to provide the instant messaging and presence functionality defined in **RFC 2779** [IMP-REQS] are specified in Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence **[XMPP-IM]**.

### 1.2. Terminology

TOC

The capitalized key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in **BCP 14, RFC 2119** [TERMS].

## 2. Generalized Architecture

TOC

### 2.1. Overview

TOC

Although XMPP is not wedded to any specific network architecture, to date it usually has been implemented via a client-server architecture wherein a client utilizing XMPP accesses a server over a **[TCP]** connection, and servers also communicate with each other over TCP connections.

The following diagram provides a high-level overview of this architecture (where "-" represents communications that use XMPP and "=" represents communications that use any other protocol).

```
C1----S1---S2---C3
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 431
Keysight, EX-1045
Page 431

```
       |
C2----+--G1===FN1===FC1
```

The symbols are as follows:

- C1, C2, C3 = XMPP clients
- S1, S2 = XMPP servers
- G1 = A gateway that translates between XMPP and the protocol(s) used on a foreign (non-XMPP) messaging network
- FN1 = A foreign messaging network
- FC1 = A client on a foreign messaging network

## 2.2. Server

TOC

A server acts as an intelligent abstraction layer for XMPP communications. Its primary responsibilities are:

- to manage connections from or sessions for other entities, in the form of **XML streams** to and from authorized clients, servers, and other entities
- to route appropriately-addressed **XML stanzas** among such entities over XML streams

Most XMPP-compliant servers also assume responsibility for the storage of data that is used by clients (e.g., contact lists for users of XMPP-based instant messaging and presence applications); in this case, the XML data is processed directly by the server itself on behalf of the client and is not routed to another entity.

## 2.3. Client

TOC

Most clients connect directly to a server over a **[TCP]** connection and use XMPP to take full advantage of the functionality provided by a server and any associated services. Multiple resources (e.g., devices or locations) MAY connect simultaneously to a server on behalf of each authorized client, with each resource differentiated by the resource identifier of an XMPP address (e.g., <node@domain/home> vs. <node@domain/work>) as defined under **Addressing Scheme**. The RECOMMENDED port for connections between a client and a server is 5222, as registered with the IANA (see **Port Numbers**).

## 2.4. Gateway

TOC

A gateway is a special-purpose server-side service whose primary function is to translate XMPP into the protocol used by a foreign (non-XMPP) messaging system, as well as to translate the return data back into XMPP. Examples are gateways to email (see **[SMTP]**), Internet Relay Chat (see **[IRC]**), SIMPLE (see **[SIMPLE]**), Short Message Service (SMS), and legacy instant messaging services such as AIM, ICQ, MSN Messenger, and Yahoo! Instant Messenger. Communications between gateways and servers, and between gateways and the foreign messaging system, are not defined in



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 432
Keysight, EX-1045
Page 432

this document.

## 2.5. Network

TOC

Because each server is identified by a network address and because server-to-server communications are a straightforward extension of the client-to-server protocol, in practice, the system consists of a network of servers that inter-communicate. Thus, for example, <juliet@example.com> is able to exchange messages, presence, and other information with <romeo@example.net>. This pattern is familiar from messaging protocols (such as **[SMTP]**) that make use of network addressing standards. Communications between any two servers are OPTIONAL. If enabled, such communications SHOULD occur over XML streams that are bound to **[TCP]** connections. The RECOMMENDED port for connections between servers is 5269, as registered with the IANA (see **Port Numbers**).

# 3. Addressing Scheme

TOC

## 3.1. Overview

TOC

An entity is anything that can be considered a network endpoint (i.e., an ID on the network) and that can communicate using XMPP. All such entities are uniquely addressable in a form that is consistent with RFC 2396 **[URI]**. For historical reasons, the address of an XMPP entity is called a Jabber Identifier or JID. A valid JID contains a set of ordered elements formed of a domain identifier, node identifier, and resource identifier.

The syntax for a JID is defined below using the Augmented Backus-Naur Form as defined in **[ABNF]**. (The IPv4address and IPv6address rules are defined in Appendix B of **[IPv6]**; the allowable character sequences that conform to the node rule are defined by the Nodeprep profile of **[STRINGPREP]** as documented in Appendix A of this memo; the allowable character sequences that conform to the resource rule are defined by the Resourceprep profile of **[STRINGPREP]** as documented in Appendix B of this memo; and the sub-domain rule makes reference to the concept of an internationalized domain label as described in **[IDNA]**.)

```
jid             = [ node "@" ] domain [ "/" resource ]
domain          = fqdn / address-literal
fqdn            = (sub-domain 1*("." sub-domain))
sub-domain      = (internationalized domain label)
address-literal = IPv4address / IPv6address
```

All JIDs are based on the foregoing structure. The most common use of this structure is to identify an instant messaging user, the server to which the user connects, and the user's connected resource (e.g., a specific client) in the form of <user@host/resource>. However, node types other than clients are possible; for example, a specific chat room offered by a multi-user chat service could be addressed as <room@service> (where



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 433
Keysight, EX-1045
Page 433

"room" is the name of the chat room and "service" is the hostname of the multi-user chat service) and a specific occupant of such a room could be addressed as <room@service/nick> (where "nick" is the occupant's room nickname). Many other JID types are possible (e.g., <domain/resource> could be a server-side script or service).

Each allowable portion of a JID (node identifier, domain identifier, and resource identifier) MUST NOT be more than 1023 bytes in length, resulting in a maximum total size (including the '@' and '/' separators) of 3071 bytes.

## 3.2. Domain Identifier

TOC

The domain identifier is the primary identifier and is the only REQUIRED element of a JID (a mere domain identifier is a valid JID). It usually represents the network gateway or "primary" server to which other entities connect for XML routing and data management capabilities. However, the entity referenced by a domain identifier is not always a server, and may be a service that is addressed as a subdomain of a server that provides functionality above and beyond the capabilities of a server (e.g., a multi-user chat service, a user directory, or a gateway to a foreign messaging system).

The domain identifier for every server or service that will communicate over a network MAY be an IP address but SHOULD be a fully qualified domain name (see **[DNS]**). A domain identifier MUST be an "internationalized domain name" as defined in **[IDNA]**, to which the **Nameprep** [NAMEPREP] profile of **stringprep** [STRINGPREP] can be applied without failing. Before comparing two domain identifiers, a server MUST (and a client SHOULD) first apply the Nameprep profile to the labels (as defined in **[IDNA]**) that make up each identifier.

## 3.3. Node Identifier

TOC

The node identifier is an optional secondary identifier placed before the domain identifier and separated from the latter by the '@' character. It usually represents the entity requesting and using network access provided by the server or gateway (i.e., a client), although it can also represent other kinds of entities (e.g., a chat room associated with a multi-user chat service). The entity represented by a node identifier is addressed within the context of a specific domain; within instant messaging and presence applications of XMPP, this address is called a "bare JID" and is of the form <node@domain>.

A node identifier MUST be formatted such that the Nodeprep profile of **[STRINGPREP]** can be applied without failing. Before comparing two node identifiers, a server MUST (and a client SHOULD) first apply the Nodeprep profile to each identifier.

## 3.4. Resource Identifier

TOC

The resource identifier is an optional tertiary identifier placed after the domain identifier and separated from the latter by the '/' character. A resource identifier may modify either a <node@domain> or a mere <domain> address. It usually represents a specific session, connection (e.g., a device or location), or object (e.g., a participant in a multi-



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 434
Keysight, EX-1045
Page 434

user chat room) belonging to the entity associated with a node identifier. A resource identifier is opaque to both servers and other clients, and is typically defined by a client implementation when it provides the information necessary to complete **Resource Binding** (although it may be generated by a server on behalf of a client), after which it is referred to as a "connected resource". An entity MAY maintain multiple connected resources simultaneously, with each connected resource differentiated by a distinct resource identifier.

A resource identifier MUST be formatted such that the Resourceprep profile of **[STRINGPREP]** can be applied without failing. Before comparing two resource identifiers, a server MUST (and a client SHOULD) first apply the Resourceprep profile to each identifier.

### 3.5. Determination of Addresses

TOC

After **SASL negotiation** and, if appropriate, **Resource Binding**, the receiving entity for a stream MUST determine the initiating entity's JID.

For server-to-server communications, the initiating entity's JID SHOULD be the authorization identity, derived from the authentication identity, as defined by the **Simple Authentication and Security Layer (SASL) specification** [SASL], if no authorization identity was specified during **SASL negotiation**.

For client-to-server communications, the "bare JID" (<node@domain>) SHOULD be the authorization identity, derived from the authentication identity, as defined in **[SASL]**, if no authorization identity was specified during **SASL negotiation**; the resource identifier portion of the "full JID" (<node@domain/resource>) SHOULD be the resource identifier negotiated by the client and server during **Resource Binding**.

The receiving entity MUST ensure that the resulting JID (including node identifier, domain identifier, resource identifier, and separator characters) conforms to the rules and formats defined earlier in this section; to meet this restriction, the receiving entity may need to replace the JID sent by the initiating entity with the canonicalized JID as determined by the receiving entity.

## 4. XML Streams

TOC

### 4.1. Overview

TOC

Two fundamental concepts make possible the rapid, asynchronous exchange of relatively small payloads of structured information between presence-aware entities: XML streams and XML stanzas. These terms are defined as follows:

Definition of XML Stream:

> An XML stream is a container for the exchange of XML elements between any two entities over a network. The start of an XML stream is denoted unambiguously by an opening XML <stream> tag (with appropriate attributes and namespace declarations), while the end of the XML stream is denoted



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 435
Keysight, EX-1045
Page 435

unambiguously by a closing XML </stream> tag. During the life of the stream, the entity that initiated it can send an unbounded number of XML elements over the stream, either elements used to negotiate the stream (e.g., to negotiate **Use of TLS** or **Use of SASL**) or XML stanzas (as defined herein, <message/>, <presence/>, or <iq/> elements qualified by the default namespace). The "initial stream" is negotiated from the initiating entity (usually a client or server) to the receiving entity (usually a server), and can be seen as corresponding to the initiating entity's "session" with the receiving entity. The initial stream enables unidirectional communication from the initiating entity to the receiving entity; in order to enable information exchange from the receiving entity to the initiating entity, the receiving entity MUST negotiate a stream in the opposite direction (the "response stream").

Definition of XML Stanza:

An XML stanza is a discrete semantic unit of structured information that is sent from one entity to another over an XML stream. An XML stanza exists at the direct child level of the root <stream/> element and is said to be well-balanced if it matches the production [43] content of **[XML]**. The start of any XML stanza is denoted unambiguously by the element start tag at depth=1 of the XML stream (e.g., <presence>), and the end of any XML stanza is denoted unambiguously by the corresponding close tag at depth=1 (e.g., </presence>). An XML stanza MAY contain child elements (with accompanying attributes, elements, and XML character data) as necessary in order to convey the desired information. The only XML stanzas defined herein are the <message/>, <presence/>, and <iq/> elements qualified by the default namespace for the stream, as described under **XML Stanzas**; an XML element sent for the purpose of **Transport Layer Security (TLS) negotiation**, **Simple Authentication and Security Layer (SASL) negotiation**, or **server dialback** is not considered to be an XML stanza.

Consider the example of a client's session with a server. In order to connect to a server, a client MUST initiate an XML stream by sending an opening <stream> tag to the server, optionally preceded by a text declaration specifying the XML version and the character encoding supported (see **Inclusion of Text Declaration**; see also **Character Encoding**). Subject to local policies and service provisioning, the server SHOULD then reply with a second XML stream back to the client, again optionally preceded by a text declaration. Once the client has completed **SASL negotiation**, the client MAY send an unbounded number of XML stanzas over the stream to any recipient on the network. When the client desires to close the stream, it simply sends a closing </stream> tag to the server (alternatively, the stream may be closed by the server), after which both the client and server SHOULD terminate the underlying connection (usually a TCP connection) as well.

Those who are accustomed to thinking of XML in a document-centric manner may wish to view a client's session with a server as consisting of two open-ended XML documents: one from the client to the server and one from the server to the client. From this perspective, the root <stream/> element can be considered the document entity for each "document", and the two "documents" are built up through the accumulation of XML stanzas sent over the two XML streams. However, this perspective is a convenience only; XMPP does not deal in documents but in XML streams and XML stanzas.

In essence, then, an XML stream acts as an envelope for all the XML stanzas sent during a session. We can represent this in a simplistic fashion as follows:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 436
Keysight, EX-1045
Page 436

```
|--------------------|
| <stream>           |
|--------------------|
| <presence>         |
|   <show/>          |
| </presence>        |
|--------------------|
| <message to='foo'> |
|   <body/>          |
| </message>         |
|--------------------|
| <iq to='bar'>      |
|   <query/>         |
| </iq>              |
|--------------------|
| ...                |
|--------------------|
| </stream>          |
|--------------------|
```

## 4.2. Binding to TCP

TOC

Although there is no necessary coupling of an XML stream to a **[TCP]** connection (e.g., two entities could connect to each other via another mechanism such as polling over **[HTTP]**), this specification defines a binding of XMPP to TCP only. In the context of client-to-server communications, a server MUST allow a client to share a single TCP connection for XML stanzas sent from client to server and from server to client. In the context of server-to-server communications, a server MUST use one TCP connection for XML stanzas sent from the server to the peer and another TCP connection (initiated by the peer) for stanzas from the peer to the server, for a total of two TCP connections.

## 4.3. Stream Security

TOC

When negotiating XML streams in XMPP 1.0, TLS SHOULD be used as defined under **Use of TLS** and SASL MUST be used as defined under **Use of SASL**. The "initial stream" (i.e., the stream from the initiating entity to the receiving entity) and the "response stream" (i.e., the stream from the receiving entity to the initiating entity) MUST be secured separately, although security in both directions MAY be established via mechanisms that provide mutual authentication. An entity SHOULD NOT attempt to send **XML Stanzas** over the stream before the stream has been authenticated, but if it does, then the other entity MUST NOT accept such stanzas and SHOULD return a <not-authorized/> stream error and then terminate both the XML stream and the underlying TCP connection; note well that this applies to XML stanzas only (i.e., <message/>, <presence/>, and <iq/> elements scoped by the default namespace) and not to XML elements used for stream negotiation (e.g., elements used to negotiate **Use of TLS** or **Use of SASL**).



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 437
Keysight, EX-1045
Page 437

### 4.4. Stream Attributes

TOC

The attributes of the stream element are as follows:

- to -- The 'to' attribute SHOULD be used only in the XML stream header from the initiating entity to the receiving entity, and MUST be set to a hostname serviced by the receiving entity. There SHOULD NOT be a 'to' attribute set in the XML stream header by which the receiving entity replies to the initiating entity; however, if a 'to' attribute is included, it SHOULD be silently ignored by the initiating entity.
- from -- The 'from' attribute SHOULD be used only in the XML stream header from the receiving entity to the initiating entity, and MUST be set to a hostname serviced by the receiving entity that is granting access to the initiating entity. There SHOULD NOT be a 'from' attribute on the XML stream header sent from the initiating entity to the receiving entity; however, if a 'from' attribute is included, it SHOULD be silently ignored by the receiving entity.
- id -- The 'id' attribute SHOULD be used only in the XML stream header from the receiving entity to the initiating entity. This attribute is a unique identifier created by the receiving entity to function as a session key for the initiating entity's streams with the receiving entity, and MUST be unique within the receiving application (normally a server). Note well that the stream ID may be security-critical and therefore MUST be both unpredictable and nonrepeating (see **[RANDOM]** for recommendations regarding randomness for security purposes). There SHOULD NOT be an 'id' attribute on the XML stream header sent from the initiating entity to the receiving entity; however, if an 'id' attribute is included, it SHOULD be silently ignored by the receiving entity.
- xml:lang -- An 'xml:lang' attribute (as defined in Section 2.12 of **[XML]**) SHOULD be included by the initiating entity on the header for the initial stream to specify the default language of any human-readable XML character data it sends over that stream. If the attribute is included, the receiving entity SHOULD remember that value as the default for both the initial stream and the response stream; if the attribute is not included, the receiving entity SHOULD use a configurable default value for both streams, which it MUST communicate in the header for the response stream. For all stanzas sent over the initial stream, if the initiating entity does not include an 'xml:lang' attribute, the receiving entity SHOULD apply the default value; if the initiating entity does include an 'xml:lang' attribute, the receiving entity MUST NOT modify or delete it (see also **xml:lang**). The value of the 'xml:lang' attribute MUST be an NMTOKEN (as defined in Section 2.3 of **[XML]**) and MUST conform to the format defined in **RFC 3066** [LANGTAGS].
- version -- The presence of the version attribute set to a value of at least "1.0" signals support for the stream-related protocols (including stream features) defined in this specification. Detailed rules regarding the generation and handling of this attribute are defined below.

We can summarize as follows:

```
         |  initiating to receiving  |  receiving to initiating
---------+---------------------------+-------------------------
to       |  hostname of receiver     |  silently ignored
```

```
from     |  silently ignored          |  hostname of receiver
id       |  silently ignored          |  session key
xml:lang |  default language          |  default language
version  |  signals XMPP 1.0 support |  signals XMPP 1.0 support
```

### 4.4.1. Version Support

TOC

The version of XMPP specified herein is "1.0"; in particular, this encapsulates the stream-related protocols (**Use of TLS**, **Use of SASL**, and **Stream Errors**), as well as the semantics of the three defined XML stanza types (<message/>, <presence/>, and <iq/>). The numbering scheme for XMPP versions is "<major>.<minor>". The major and minor numbers MUST be treated as separate integers and each number MAY be incremented higher than a single digit. Thus, "XMPP 2.4" would be a lower version than "XMPP 2.13", which in turn would be lower than "XMPP 12.3". Leading zeros (e.g., "XMPP 6.01") MUST be ignored by recipients and MUST NOT be sent.

The major version number should be incremented only if the stream and stanza formats or required actions have changed so dramatically that an older version entity would not be able to interoperate with a newer version entity if it simply ignored the elements and attributes it did not understand and took the actions specified in the older specification. The minor version number indicates new capabilities, and MUST be ignored by an entity with a smaller minor version number, but used for informational purposes by the entity with the larger minor version number. For example, a minor version number might indicate the ability to process a newly defined value of the 'type' attribute for message, presence, or IQ stanzas; the entity with the larger minor version number would simply note that its correspondent would not be able to understand that value of the 'type' attribute and therefore would not send it.

The following rules apply to the generation and handling of the 'version' attribute within stream headers by implementations:

1. The initiating entity MUST set the value of the 'version' attribute on the initial stream header to the highest version number it supports (e.g., if the highest version number it supports is that defined in this specification, it MUST set the value to "1.0").
2. The receiving entity MUST set the value of the 'version' attribute on the response stream header to either the value supplied by the initiating entity or the highest version number supported by the receiving entity, whichever is lower. The receiving entity MUST perform a numeric comparison on the major and minor version numbers, not a string match on "<major>.<minor>".
3. If the version number included in the response stream header is at least one major version lower than the version number included in the initial stream header and newer version entities cannot interoperate with older version entities as described above, the initiating entity SHOULD generate an <unsupported-version/> stream error and terminate the XML stream and underlying TCP connection.
4. If either entity receives a stream header with no 'version' attribute, the entity MUST consider the version supported by the other entity to be "0.0" and SHOULD NOT include a 'version' attribute in the stream header it sends in reply.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 439
Keysight, EX-1045
Page 439

### 4.5. Namespace Declarations

TOC

The stream element MUST possess both a streams namespace declaration and a default namespace declaration (as "namespace declaration" is defined in the **XML namespaces specification** [XML-NAMES]). For detailed information regarding the streams namespace and default namespace, see **Namespace Names and Prefixes**.

### 4.6. Stream Features

TOC

If the initiating entity includes the 'version' attribute set to a value of at least "1.0" in the initial stream header, the receiving entity MUST send a <features/> child element (prefixed by the streams namespace prefix) to the initiating entity in order to announce any stream-level features that can be negotiated (or capabilities that otherwise need to be advertised). Currently, this is used only to advertise **Use of TLS**, **Use of SASL**, and **Resource Binding** as defined herein, and for Session Establishment as defined in **[XMPP-IM]**; however, the stream features functionality could be used to advertise other negotiable features in the future. If an entity does not understand or support some features, it SHOULD silently ignore them. If one or more security features (e.g., TLS and SASL) need to be successfully negotiated before a non-security-related feature (e.g., Resource Binding) can be offered, the non-security-related feature SHOULD NOT be included in the stream features that are advertised before the relevant security features have been negotiated.

### 4.7. Stream Errors

TOC

The root stream element MAY contain an <error/> child element that is prefixed by the streams namespace prefix. The error child MUST be sent by a compliant entity (usually a server rather than a client) if it perceives that a stream-level error has occurred.

#### 4.7.1. Rules

TOC

The following rules apply to stream-level errors:

- It is assumed that all stream-level errors are unrecoverable; therefore, if an error occurs at the level of the stream, the entity that detects the error MUST send a stream error to the other entity, send a closing </stream> tag, and terminate the underlying TCP connection.
- If the error occurs while the stream is being set up, the receiving entity MUST still send the opening <stream> tag, include the <error/> element as a child of the stream element, send the closing </stream> tag, and terminate the underlying TCP connection. In this case, if the initiating entity provides an unknown host in the 'to' attribute (or provides no 'to' attribute at all), the server SHOULD provide the server's authoritative hostname in the 'from' attribute of the stream header sent before termination.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 440
Keysight, EX-1045
Page 440

### 4.7.2. Syntax

TOC

The syntax for stream errors is as follows:

```
<stream:error>
  <defined-condition xmlns='urn:ietf:params:xml:ns:xmpp-streams'/>
  <text xmlns='urn:ietf:params:xml:ns:xmpp-streams'
        xml:lang='langcode'>
    OPTIONAL descriptive text
  </text>
  [OPTIONAL application-specific condition element]
</stream:error>
```

The <error/> element:

- MUST contain a child element corresponding to one of the defined stanza error conditions defined below; this element MUST be qualified by the 'urn:ietf:params:xml:ns:xmpp-streams' namespace
- MAY contain a <text/> child containing XML character data that describes the error in more detail; this element MUST be qualified by the 'urn:ietf:params:xml:ns:xmpp-streams' namespace and SHOULD possess an 'xml:lang' attribute specifying the natural language of the XML character data
- MAY contain a child element for an application-specific error condition; this element MUST be qualified by an application-defined namespace, and its structure is defined by that namespace

The <text/> element is OPTIONAL. If included, it SHOULD be used only to provide descriptive or diagnostic information that supplements the meaning of a defined condition or application-specific condition. It SHOULD NOT be interpreted programmatically by an application. It SHOULD NOT be used as the error message presented to a user, but MAY be shown in addition to the error message associated with the included condition element (or elements).

### 4.7.3. Defined Conditions

TOC

The following stream-level error conditions are defined:

- <bad-format/> -- the entity has sent XML that cannot be processed; this error MAY be used instead of the more specific XML-related errors, such as <bad-namespace-prefix/>, <invalid-xml/>, <restricted-xml/>, <unsupported-encoding/>, and <xml-not-well-formed/>, although the more specific errors are preferred.
- <bad-namespace-prefix/> -- the entity has sent a namespace prefix that is unsupported, or has sent no namespace prefix on an element that requires such a prefix (see **XML Namespace Names and Prefixes**).
- <conflict/> -- the server is closing the active stream for this entity because a new stream has been initiated that conflicts with the existing stream.
- <connection-timeout/> -- the entity has not generated any traffic over the



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 441
Keysight, EX-1045
Page 441

stream for some period of time (configurable according to a local service policy).

- <host-gone/> -- the value of the 'to' attribute provided by the initiating entity in the stream header corresponds to a hostname that is no longer hosted by the server.
- <host-unknown/> -- the value of the 'to' attribute provided by the initiating entity in the stream header does not correspond to a hostname that is hosted by the server.
- <improper-addressing/> -- a stanza sent between two servers lacks a 'to' or 'from' attribute (or the attribute has no value).
- <internal-server-error/> -- the server has experienced a misconfiguration or an otherwise-undefined internal error that prevents it from servicing the stream.
- <invalid-from/> -- the JID or hostname provided in a 'from' address does not match an authorized JID or validated domain negotiated between servers via SASL or dialback, or between a client and a server via authentication and resource binding.
- <invalid-id/> -- the stream ID or dialback ID is invalid or does not match an ID previously provided.
- <invalid-namespace/> -- the streams namespace name is something other than "http://etherx.jabber.org/streams" or the dialback namespace name is something other than "jabber:server:dialback" (see **XML Namespace Names and Prefixes**).
- <invalid-xml/> -- the entity has sent invalid XML over the stream to a server that performs validation (see **Validation**).
- <not-authorized/> -- the entity has attempted to send data before the stream has been authenticated, or otherwise is not authorized to perform an action related to stream negotiation; the receiving entity MUST NOT process the offending stanza before sending the stream error.
- <policy-violation/> -- the entity has violated some local service policy; the server MAY choose to specify the policy in the <text/> element or an application-specific condition element.
- <remote-connection-failed/> -- the server is unable to properly connect to a remote entity that is required for authentication or authorization.
- <resource-constraint/> -- the server lacks the system resources necessary to service the stream.
- <restricted-xml/> -- the entity has attempted to send restricted XML features such as a comment, processing instruction, DTD, entity reference, or unescaped character (see **Restrictions**).
- <see-other-host/> -- the server will not provide service to the initiating entity but is redirecting traffic to another host; the server SHOULD specify the alternate hostname or IP address (which MUST be a valid domain identifier) as the XML character data of the <see-other-host/> element.
- <system-shutdown/> -- the server is being shut down and all active streams are being closed.
- <undefined-condition/> -- the error condition is not one of those defined by the other conditions in this list; this error condition SHOULD be used only in conjunction with an application-specific condition.
- <unsupported-encoding/> -- the initiating entity has encoded the stream in an encoding that is not supported by the server (see **Character Encoding**).
- <unsupported-stanza-type/> -- the initiating entity has sent a first-level child of the stream that is not supported by the server.
- <unsupported-version/> -- the value of the 'version' attribute provided by



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 442
Keysight, EX-1045
Page 442

the initiating entity in the stream header specifies a version of XMPP that is not supported by the server; the server MAY specify the version(s) it supports in the <text/> element.

- <xml-not-well-formed/> -- the initiating entity has sent XML that is not well-formed as defined by **[XML]**.

### 4.7.4. Application-Specific Conditions

TOC

As noted, an application MAY provide application-specific stream error information by including a properly-namespaced child in the error element. The application-specific element SHOULD supplement or further qualify a defined element. Thus the <error/> element will contain two or three child elements:

```
<stream:error>
  <xml-not-well-formed
      xmlns='urn:ietf:params:xml:ns:xmpp-streams'/>
  <text xml:lang='en' xmlns='urn:ietf:params:xml:ns:xmpp-streams'>
    Some special application diagnostic information!
  </text>
  <escape-your-data xmlns='application-ns'/>
</stream:error>
</stream:stream>
```

## 4.8. Simplified Stream Examples

TOC

This section contains two simplified examples of a stream-based "session" of a client on a server (where the "C" lines are sent from the client to the server, and the "S" lines are sent from the server to the client); these examples are included for the purpose of illustrating the concepts introduced thus far.

A basic "session":

```
C: <?xml version='1.0'?>
   <stream:stream
       to='example.com'
       xmlns='jabber:client'
       xmlns:stream='http://etherx.jabber.org/streams'
       version='1.0'>
S: <?xml version='1.0'?>
   <stream:stream
       from='example.com'
       id='someid'
       xmlns='jabber:client'
       xmlns:stream='http://etherx.jabber.org/streams'
       version='1.0'>
...  encryption, authentication, and resource binding ...
C:   <message from='juliet@example.com'
              to='romeo@example.net'
              xml:lang='en'>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 443
Keysight, EX-1045
Page 443

```
C:     <body>Art thou not Romeo, and a Montague?</body>
C:   </message>
S:   <message from='romeo@example.net'
               to='juliet@example.com'
               xml:lang='en'>
S:     <body>Neither, fair saint, if either thee dislike.</body>
S:   </message>
C: </stream:stream>
S: </stream:stream>
```

A "session" gone bad:

```
C: <?xml version='1.0'?>
   <stream:stream
       to='example.com'
       xmlns='jabber:client'
       xmlns:stream='http://etherx.jabber.org/streams'
       version='1.0'>
S: <?xml version='1.0'?>
   <stream:stream
       from='example.com'
       id='someid'
       xmlns='jabber:client'
       xmlns:stream='http://etherx.jabber.org/streams'
       version='1.0'>
...  encryption, authentication, and resource binding ...
C: <message xml:lang='en'>
     <body>Bad XML, no closing body tag!
   </message>
S: <stream:error>
    <xml-not-well-formed
        xmlns='urn:ietf:params:xml:ns:xmpp-streams'/>
   </stream:error>
S: </stream:stream>
```

## 5. Use of TLS

TOC

### 5.1. Overview

TOC

XMPP includes a method for securing the stream from tampering and eavesdropping. This channel encryption method makes use of the **Transport Layer Security (TLS) protocol** [TLS], along with a "STARTTLS" extension that is modelled after similar extensions for the **IMAP** [IMAP], **POP3** [POP3], and **ACAP** [ACAP] protocols as described in **RFC 2595** [USINGTLS]. The namespace name for the STARTTLS extension is 'urn:ietf:params:xml:ns:xmpp-tls'.

An administrator of a given domain MAY require the use of TLS for client-to-server communications, server-to-server communications, or both. Clients SHOULD use TLS to secure the streams prior to attempting the completion of **SASL negotiation**, and servers SHOULD use TLS between two domains for the purpose of securing server-to-



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 444
Keysight, EX-1045
Page 444

server communications.

The following rules apply:

1. An initiating entity that complies with this specification MUST include the 'version' attribute set to a value of "1.0" in the initial stream header.
2. If the TLS negotiation occurs between two servers, communications MUST NOT proceed until the Domain Name System (DNS) hostnames asserted by the servers have been resolved (see **Server-to-Server Communications**).
3. When a receiving entity that complies with this specification receives an initial stream header that includes the 'version' attribute set to a value of at least "1.0", after sending a stream header in reply (including the version flag), it MUST include a <starttls/> element (qualified by the 'urn:ietf:params:xml:ns:xmpp-tls' namespace) along with the list of other stream features it supports.
4. If the initiating entity chooses to use TLS, TLS negotiation MUST be completed before proceeding to SASL negotiation; this order of negotiation is required to help safeguard authentication information sent during SASL negotiation, as well as to make it possible to base the use of the SASL EXTERNAL mechanism on a certificate provided during prior TLS negotiation.
5. During TLS negotiation, an entity MUST NOT send any white space characters (matching production [3] content of **[XML]**) within the root stream element as separators between elements (any white space characters shown in the TLS examples below are included for the sake of readability only); this prohibition helps to ensure proper security layer byte precision.
6. The receiving entity MUST consider the TLS negotiation to have begun immediately after sending the closing ">" character of the <proceed/> element. The initiating entity MUST consider the TLS negotiation to have begun immediately after receiving the closing ">" character of the <proceed/> element from the receiving entity.
7. The initiating entity MUST validate the certificate presented by the receiving entity; see **Certificate Validation** regarding certificate validation procedures.
8. Certificates MUST be checked against the hostname as provided by the initiating entity (e.g., a user), not the hostname as resolved via the Domain Name System; e.g., if the user specifies a hostname of "example.com" but a **DNS SRV** [SRV] lookup returned "im.example.com", the certificate MUST be checked as "example.com". If a JID for any kind of XMPP entity (e.g., client or server) is represented in a certificate, it MUST be represented as a UTF8String within an otherName entity inside the subjectAltName, using the **[ASN.1]** Object Identifier "id-on-xmppAddr" specified in **Section 5.1.1** of this document.
9. If the TLS negotiation is successful, the receiving entity MUST discard any knowledge obtained in an insecure manner from the initiating entity before TLS takes effect.
10. If the TLS negotiation is successful, the initiating entity MUST discard any knowledge obtained in an insecure manner from the receiving entity before TLS takes effect.
11. If the TLS negotiation is successful, the receiving entity MUST NOT offer the STARTTLS extension to the initiating entity along with the other stream features that are offered when the stream is restarted.
12. If the TLS negotiation is successful, the initiating entity MUST continue with SASL negotiation.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 445
Keysight, EX-1045
Page 445

13. If the TLS negotiation results in failure, the receiving entity MUST terminate both the XML stream and the underlying TCP connection.
14. See **Mandatory-to-Implement Technologies** regarding mechanisms that MUST be supported.

### 5.1.1. ASN.1 Object Identifier for XMPP Address

TOC

The **[ASN.1]** Object Identifier "id-on-xmppAddr" described above is defined as follows:

```
id-pkix OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
        dod(6) internet(1) security(5) mechanisms(5) pkix(7) }

id-on  OBJECT IDENTIFIER ::= { id-pkix 8 }  -- other name forms

id-on-xmppAddr  OBJECT IDENTIFIER ::= { id-on 5 }

XmppAddr ::= UTF8String
```

This Object Identifier MAY also be represented in dotted display format as "1.3.6.1.5.5.7.8.5".

## 5.2. Narrative

TOC

When an initiating entity secures a stream with a receiving entity using TLS, the steps involved are as follows:

1. The initiating entity opens a TCP connection and initiates the stream by sending the opening XML stream header to the receiving entity, including the 'version' attribute set to a value of at least "1.0".
2. The receiving entity responds by opening a TCP connection and sending an XML stream header to the initiating entity, including the 'version' attribute set to a value of at least "1.0".
3. The receiving entity offers the STARTTLS extension to the initiating entity by including it with the list of other supported stream features (if TLS is required for interaction with the receiving entity, it SHOULD signal that fact by including a <required/> element as a child of the <starttls/> element).
4. The initiating entity issues the STARTTLS command (i.e., a <starttls/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-tls' namespace) to instruct the receiving entity that it wishes to begin a TLS negotiation to secure the stream.
5. The receiving entity MUST reply with either a <proceed/> element or a <failure/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-tls' namespace. If the failure case occurs, the receiving entity MUST terminate both the XML stream and the underlying TCP connection. If the proceed case occurs, the entities MUST attempt to complete the TLS negotiation over the TCP connection and MUST NOT send any further XML data until the TLS negotiation is complete.
6. The initiating entity and receiving entity attempt to complete a TLS



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 446
Keysight, EX-1045
Page 446

negotiation in accordance with **[TLS]**.

7. If the TLS negotiation is unsuccessful, the receiving entity MUST terminate the TCP connection. If the TLS negotiation is successful, the initiating entity MUST initiate a new stream by sending an opening XML stream header to the receiving entity (it is not necessary to send a closing </stream> tag first, since the receiving entity and initiating entity MUST consider the original stream to be closed upon successful TLS negotiation).
8. Upon receiving the new stream header from the initiating entity, the receiving entity MUST respond by sending a new XML stream header to the initiating entity along with the available features (but not including the STARTTLS feature).

---

### 5.3. Client-to-Server Example

TOC

The following example shows the data flow for a client securing a stream using STARTTLS (note: the alternate steps shown below are provided to illustrate the protocol for failure cases; they are not exhaustive and would not necessarily be triggered by the data sent in the example).

Step 1: Client initiates stream to server:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    to='example.com'
    version='1.0'>
```

Step 2: Server responds by sending a stream tag to client:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    id='c2s_123'
    from='example.com'
    version='1.0'>
```

Step 3: Server sends the STARTTLS extension to client along with authentication mechanisms and any other stream features:

```
<stream:features>
  <starttls xmlns='urn:ietf:params:xml:ns:xmpp-tls'>
    <required/>
  </starttls>
  <mechanisms xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
    <mechanism>DIGEST-MD5</mechanism>
    <mechanism>PLAIN</mechanism>
  </mechanisms>
</stream:features>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 447
Keysight, EX-1045
Page 447

Step 4: Client sends the STARTTLS command to server:

```
<starttls xmlns='urn:ietf:params:xml:ns:xmpp-tls'/>
```

Step 5: Server informs client that it is allowed to proceed:

```
<proceed xmlns='urn:ietf:params:xml:ns:xmpp-tls'/>
```

Step 5 (alt): Server informs client that TLS negotiation has failed and closes both stream and TCP connection:

```
<failure xmlns='urn:ietf:params:xml:ns:xmpp-tls'/>
</stream:stream>
```

Step 6: Client and server attempt to complete TLS negotiation over the existing TCP connection.

Step 7: If TLS negotiation is successful, client initiates a new stream to server:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    to='example.com'
    version='1.0'>
```

Step 7 (alt): If TLS negotiation is unsuccessful, server closes TCP connection.

Step 8: Server responds by sending a stream header to client along with any available stream features:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    from='example.com'
    id='c2s_234'
    version='1.0'>
<stream:features>
  <mechanisms xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
    <mechanism>DIGEST-MD5</mechanism>
    <mechanism>PLAIN</mechanism>
    <mechanism>EXTERNAL</mechanism>
  </mechanisms>
</stream:features>
```

Step 9: Client continues with **SASL negotiation**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 448
Keysight, EX-1045
Page 448

### 5.4. Server-to-Server Example

TOC

The following example shows the data flow for two servers securing a stream using STARTTLS (note: the alternate steps shown below are provided to illustrate the protocol for failure cases; they are not exhaustive and would not necessarily be triggered by the data sent in the example).

Step 1: Server1 initiates stream to Server2:

```
<stream:stream
    xmlns='jabber:server'
    xmlns:stream='http://etherx.jabber.org/streams'
    to='example.com'
    version='1.0'>
```

Step 2: Server2 responds by sending a stream tag to Server1:

```
<stream:stream
    xmlns='jabber:server'
    xmlns:stream='http://etherx.jabber.org/streams'
    from='example.com'
    id='s2s_123'
    version='1.0'>
```

Step 3: Server2 sends the STARTTLS extension to Server1 along with authentication mechanisms and any other stream features:

```
<stream:features>
  <starttls xmlns='urn:ietf:params:xml:ns:xmpp-tls'>
    <required/>
  </starttls>
  <mechanisms xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
    <mechanism>DIGEST-MD5</mechanism>
    <mechanism>KERBEROS_V4</mechanism>
  </mechanisms>
</stream:features>
```

Step 4: Server1 sends the STARTTLS command to Server2:

```
<starttls xmlns='urn:ietf:params:xml:ns:xmpp-tls'/>
```

Step 5: Server2 informs Server1 that it is allowed to proceed:

```
<proceed xmlns='urn:ietf:params:xml:ns:xmpp-tls'/>
```

Step 5 (alt): Server2 informs Server1 that TLS negotiation has failed and closes stream:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 449
Keysight, EX-1045
Page 449

```
<failure xmlns='urn:ietf:params:xml:ns:xmpp-tls'/>
</stream:stream>
```

Step 6: Server1 and Server2 attempt to complete TLS negotiation via TCP.

Step 7: If TLS negotiation is successful, Server1 initiates a new stream to Server2:

```
<stream:stream
    xmlns='jabber:server'
    xmlns:stream='http://etherx.jabber.org/streams'
    to='example.com'
    version='1.0'>
```

Step 7 (alt): If TLS negotiation is unsuccessful, Server2 closes TCP connection.

Step 8: Server2 responds by sending a stream header to Server1 along with any available stream features:

```
<stream:stream
    xmlns='jabber:server'
    xmlns:stream='http://etherx.jabber.org/streams'
    from='example.com'
    id='s2s_234'
    version='1.0'>
<stream:features>
  <mechanisms xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
    <mechanism>DIGEST-MD5</mechanism>
    <mechanism>KERBEROS_V4</mechanism>
    <mechanism>EXTERNAL</mechanism>
  </mechanisms>
</stream:features>
```

Step 9: Server1 continues with **SASL negotiation**.

## 6. Use of SASL

TOC

### 6.1. Overview

TOC

XMPP includes a method for authenticating a stream by means of an XMPP-specific profile of the **Simple Authentication and Security Layer (SASL) protocol** [SASL]. SASL provides a generalized method for adding authentication support to connection-based protocols, and XMPP uses a generic XML namespace profile for SASL that conforms to the profiling requirements of **[SASL]**.

The following rules apply:

1. If the SASL negotiation occurs between two servers, communications MUST



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 450
Keysight, EX-1045
Page 450

NOT proceed until the Domain Name System (DNS) hostnames asserted by the servers have been resolved (see **Server-to-Server Communications**).
2. If the initiating entity is capable of SASL negotiation, it MUST include the 'version' attribute set to a value of at least "1.0" in the initial stream header.
3. If the receiving entity is capable of SASL negotiation, it MUST advertise one or more authentication mechanisms within a <mechanisms/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-sasl' namespace in reply to the opening stream tag received from the initiating entity (if the opening stream tag included the 'version' attribute set to a value of at least "1.0").
4. During SASL negotiation, an entity MUST NOT send any white space characters (matching production [3] content of **[XML]**) within the root stream element as separators between elements (any white space characters shown in the SASL examples below are included for the sake of readability only); this prohibition helps to ensure proper security layer byte precision.
5. Any XML character data contained within the XML elements used during SASL negotiation MUST be encoded using base64, where the encoding adheres to the definition in Section 3 of **RFC 3548** [BASE64].
6. If provision of a "simple username" is supported by the selected SASL mechanism (e.g., this is supported by the DIGEST-MD5 and CRAM-MD5 mechanisms but not by the EXTERNAL and GSSAPI mechanisms), during authentication the initiating entity SHOULD provide as the simple username its sending domain (IP address or fully qualified domain name as contained in a domain identifier) in the case of server-to-server communications or its registered account name (user or node name as contained in an XMPP node identifier) in the case of client-to-server communications.
7. If the initiating entity wishes to act on behalf of another entity and the selected SASL mechanism supports transmission of an authorization identity, the initiating entity MUST provide an authorization identity during SASL negotiation. If the initiating entity does not wish to act on behalf of another entity, it MUST NOT provide an authorization identity. As specified in **[SASL]**, the initiating entity MUST NOT provide an authorization identity unless the authorization identity is different from the default authorization identity derived from the authentication identity, as described in **[SASL]**. If provided, the value of the authorization identity MUST be of the form <domain> (i.e., a domain identifier only) for servers and of the form <node@domain> (i.e., node identifier and domain identifier) for clients.
8. Upon successful SASL negotiation that involves negotiation of a security layer, the receiving entity MUST discard any knowledge obtained from the initiating entity which was not obtained from the SASL negotiation itself.
9. Upon successful SASL negotiation that involves negotiation of a security layer, the initiating entity MUST discard any knowledge obtained from the receiving entity which was not obtained from the SASL negotiation itself.
10. See **Mandatory-to-Implement Technologies** regarding mechanisms that MUST be supported.

---

### 6.2. Narrative

TOC

When an initiating entity authenticates with a receiving entity using SASL, the steps involved are as follows:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 451
Keysight, EX-1045
Page 451

1. The initiating entity requests SASL authentication by including the 'version' attribute in the opening XML stream header sent to the receiving entity, with the value set to "1.0".
2. After sending an XML stream header in reply, the receiving entity advertises a list of available SASL authentication mechanisms; each of these is a <mechanism/> element included as a child within a <mechanisms/> container element qualified by the 'urn:ietf:params:xml:ns:xmpp-sasl' namespace, which in turn is a child of a <features/> element in the streams namespace. If **Use of TLS** needs to be established before a particular authentication mechanism may be used, the receiving entity MUST NOT provide that mechanism in the list of available SASL authentication mechanisms prior to TLS negotiation. If the initiating entity presents a valid certificate during prior TLS negotiation, the receiving entity SHOULD offer the SASL EXTERNAL mechanism to the initiating entity during SASL negotiation (refer to **[SASL]**), although the EXTERNAL mechanism MAY be offered under other circumstances as well.
3. The initiating entity selects a mechanism by sending an <auth/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-sasl' namespace to the receiving entity and including an appropriate value for the 'mechanism' attribute. This element MAY contain XML character data (in SASL terminology, the "initial response") if the mechanism supports or requires it; if the initiating entity needs to send a zero-length initial response, it MUST transmit the response as a single equals sign ("="), which indicates that the response is present but contains no data.
4. If necessary, the receiving entity challenges the initiating entity by sending a <challenge/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-sasl' namespace to the initiating entity; this element MAY contain XML character data (which MUST be computed in accordance with the definition of the SASL mechanism chosen by the initiating entity).
5. The initiating entity responds to the challenge by sending a <response/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-sasl' namespace to the receiving entity; this element MAY contain XML character data (which MUST be computed in accordance with the definition of the SASL mechanism chosen by the initiating entity).
6. If necessary, the receiving entity sends more challenges and the initiating entity sends more responses.

This series of challenge/response pairs continues until one of three things happens:

1. The initiating entity aborts the handshake by sending an <abort/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-sasl' namespace to the receiving entity. Upon receiving an <abort/> element, the receiving entity SHOULD allow a configurable but reasonable number of retries (at least 2), after which it MUST terminate the TCP connection; this enables the initiating entity (e.g., an end-user client) to tolerate incorrectly-provided credentials (e.g., a mistyped password) without being forced to reconnect.
2. The receiving entity reports failure of the handshake by sending a <failure/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-sasl' namespace to the initiating entity (the particular cause of failure SHOULD be communicated in an appropriate child element of the <failure/> element as defined under **SASL Errors**). If the failure case occurs, the receiving entity SHOULD allow a configurable but reasonable number of retries (at least 2), after which it MUST terminate the TCP connection; this enables the initiating entity (e.g., an end-user client) to tolerate incorrectly-provided credentials



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 452
Keysight, EX-1045
Page 452

(e.g., a mistyped password) without being forced to reconnect.

3. The receiving entity reports success of the handshake by sending a <success/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-sasl' namespace to the initiating entity; this element MAY contain XML character data (in SASL terminology, "additional data with success") if required by the chosen SASL mechanism. Upon receiving the <success/> element, the initiating entity MUST initiate a new stream by sending an opening XML stream header to the receiving entity (it is not necessary to send a closing </stream> tag first, since the receiving entity and initiating entity MUST consider the original stream to be closed upon sending or receiving the <success/> element). Upon receiving the new stream header from the initiating entity, the receiving entity MUST respond by sending a new XML stream header to the initiating entity, along with any available features (but not including the STARTTLS and SASL features) or an empty <features/> element (to signify that no additional features are available); any such additional features not defined herein MUST be defined by the relevant extension to XMPP.

## 6.3. SASL Definition

TOC

The profiling requirements of **[SASL]** require that the following information be supplied by a protocol definition:

service name:
: "xmpp"

initiation sequence:
: After the initiating entity provides an opening XML stream header and the receiving entity replies in kind, the receiving entity provides a list of acceptable authentication methods. The initiating entity chooses one method from the list and sends it to the receiving entity as the value of the 'mechanism' attribute possessed by an <auth/> element, optionally including an initial response to avoid a round trip.

exchange sequence:
: Challenges and responses are carried through the exchange of <challenge/> elements from receiving entity to initiating entity and <response/> elements from initiating entity to receiving entity. The receiving entity reports failure by sending a <failure/> element and success by sending a <success/> element; the initiating entity aborts the exchange by sending an <abort/> element. Upon successful negotiation, both sides consider the original XML stream to be closed and new stream headers are sent by both entities.

security layer negotiation:
: The security layer takes effect immediately after sending the closing ">" character of the <success/> element for the receiving entity, and immediately after receiving the closing ">" character of the <success/> element for the initiating entity. The order of layers is first **[TCP]**, then **[TLS]**, then **[SASL]**, then XMPP.

use of the authorization identity:
: The authorization identity may be used by xmpp to denote the non-default <node@domain> of a client or the sending <domain> of a server.

https://web.archive.org/web/20061010092431/https://xmpp.org/rfcs/rfc3920.html

### 6.4. SASL Errors

TOC

The following SASL-related error conditions are defined:

- <aborted/> -- The receiving entity acknowledges an <abort/> element sent by the initiating entity; sent in reply to the <abort/> element.
- <incorrect-encoding/> -- The data provided by the initiating entity could not be processed because the **[BASE64]** encoding is incorrect (e.g., because the encoding does not adhere to the definition in Section 3 of **[BASE64]**); sent in reply to a <response/> element or an <auth/> element with initial response data.
- <invalid-authzid/> -- The authzid provided by the initiating entity is invalid, either because it is incorrectly formatted or because the initiating entity does not have permissions to authorize that ID; sent in reply to a <response/> element or an <auth/> element with initial response data.
- <invalid-mechanism/> -- The initiating entity did not provide a mechanism or requested a mechanism that is not supported by the receiving entity; sent in reply to an <auth/> element.
- <mechanism-too-weak/> -- The mechanism requested by the initiating entity is weaker than server policy permits for that initiating entity; sent in reply to a <response/> element or an <auth/> element with initial response data.
- <not-authorized/> -- The authentication failed because the initiating entity did not provide valid credentials (this includes but is not limited to the case of an unknown username); sent in reply to a <response/> element or an <auth/> element with initial response data.
- <temporary-auth-failure/> -- The authentication failed because of a temporary error condition within the receiving entity; sent in reply to an <auth/> element or <response/> element.

---

### 6.5. Client-to-Server Example

TOC

The following example shows the data flow for a client authenticating with a server using SASL, normally after successful TLS negotiation (note: the alternate steps shown below are provided to illustrate the protocol for failure cases; they are not exhaustive and would not necessarily be triggered by the data sent in the example).

Step 1: Client initiates stream to server:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    to='example.com'
    version='1.0'>
```

Step 2: Server responds with a stream tag sent to client:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 454
Keysight, EX-1045
Page 454

```
    id='c2s_234'
    from='example.com'
    version='1.0'>
```

Step 3: Server informs client of available authentication mechanisms:

```
<stream:features>
  <mechanisms xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
    <mechanism>DIGEST-MD5</mechanism>
    <mechanism>PLAIN</mechanism>
  </mechanisms>
</stream:features>
```

Step 4: Client selects an authentication mechanism:

```
<auth xmlns='urn:ietf:params:xml:ns:xmpp-sasl'
      mechanism='DIGEST-MD5'/>
```

Step 5: Server sends a **[BASE64]** encoded challenge to client:

```
<challenge xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
cmVhbG09InNvbWVyZWFsbSIsbm9uY2U9Ik9BNk1HOXRFUUdtMmhoIixxb3A9ImF1dGgi
LGNoYXJzZXQ9dXRmLTgsYWxnb3JpdGhtPW1kNS1zZXNzCg==
</challenge>
```

The decoded challenge is:

```
realm="somerealm",nonce="OA6MG9tEQGm2hh",\
qop="auth",charset=utf-8,algorithm=md5-sess
```

Step 5 (alt): Server returns error to client:

```
<failure xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
  <incorrect-encoding/>
</failure>
</stream:stream>
```

Step 6: Client sends a **[BASE64]** encoded response to the challenge:

```
<response xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
dXNlcm5hbWU9InNvbWVub2RlIixyZWFsbT0ic29tZXJlYWxtIixub25jZT0i
T0E2TUc5dEVRR20yaGgiLGNub25jZT0iT0E2TUhYaDZWcVRyUmsiLG5jPTAw
MDAwMDAxLHFvcD1hdXRoLGRpZ2VzdC11cmk9InhtcHAvZXhhbXBsZS5jb20i
LHJlc3BvbnNlPWQzODhkYWQ5MGQ0YmJkNzYwYTE1MjMyMWYyMTQzYWY3LGNo
YXJzZXQ9dXRmLTgK
</response>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 455
Keysight, EX-1045
Page 455

The decoded response is:

```
username="somenode",realm="somerealm",\
nonce="OA6MG9tEQGm2hh",cnonce="OA6MHXh6VqTrRk",\
nc=00000001,qop=auth,digest-uri="xmpp/example.com",\
response=d388dad90d4bbd760a152321f2143af7,charset=utf-8
```

Step 7: Server sends another **[BASE64]** encoded challenge to client:

```
<challenge xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
cnNwYXV0aD1lYTQwZjYwMzM1YzQyN2I1NTI3Yjg0ZGJhYmNkZmZmZAo=
</challenge>
```

The decoded challenge is:

```
rspauth=ea40f60335c427b5527b84dbabcdfffd
```

Step 7 (alt): Server returns error to client:

```
<failure xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
  <temporary-auth-failure/>
</failure>
</stream:stream>
```

Step 8: Client responds to the challenge:

```
<response xmlns='urn:ietf:params:xml:ns:xmpp-sasl'/>
```

Step 9: Server informs client of successful authentication:

```
<success xmlns='urn:ietf:params:xml:ns:xmpp-sasl'/>
```

Step 9 (alt): Server informs client of failed authentication:

```
<failure xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
  <temporary-auth-failure/>
</failure>
</stream:stream>
```

Step 10: Client initiates a new stream to server:

```
<stream:stream
    xmlns='jabber:client'
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 456
Keysight, EX-1045
Page 456

```
    xmlns:stream='http://etherx.jabber.org/streams'
    to='example.com'
    version='1.0'>
```

Step 11: Server responds by sending a stream header to client along with any additional features (or an empty features element):

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    id='c2s_345'
    from='example.com'
    version='1.0'>
<stream:features>
  <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'/>
  <session xmlns='urn:ietf:params:xml:ns:xmpp-session'/>
</stream:features>
```

## 6.6. Server-to-Server Example

TOC

The following example shows the data flow for a server authenticating with another server using SASL, normally after successful TLS negotiation (note: the alternate steps shown below are provided to illustrate the protocol for failure cases; they are not exhaustive and would not necessarily be triggered by the data sent in the example).

Step 1: Server1 initiates stream to Server2:

```
<stream:stream
    xmlns='jabber:server'
    xmlns:stream='http://etherx.jabber.org/streams'
    to='example.com'
    version='1.0'>
```

Step 2: Server2 responds with a stream tag sent to Server1:

```
<stream:stream
    xmlns='jabber:server'
    xmlns:stream='http://etherx.jabber.org/streams'
    from='example.com'
    id='s2s_234'
    version='1.0'>
```

Step 3: Server2 informs Server1 of available authentication mechanisms:

```
<stream:features>
  <mechanisms xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
    <mechanism>DIGEST-MD5</mechanism>
    <mechanism>KERBEROS_V4</mechanism>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 457
Keysight, EX-1045
Page 457

```
  </mechanisms>
</stream:features>
```

Step 4: Server1 selects an authentication mechanism:

```
<auth xmlns='urn:ietf:params:xml:ns:xmpp-sasl'
      mechanism='DIGEST-MD5'/>
```

Step 5: Server2 sends a **[BASE64]** encoded challenge to Server1:

```
<challenge xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
cmVhbG09InNvbWVyZWFsbSIsbm9uY2U9Ik9BNk1HOXRFUUdtMmhoIixxb3A9
ImF1dGgiLGNoYXJzZXQ9dXRmLTgsYWxnb3JpdGhtPW1kNS1zZXNz
</challenge>
```

The decoded challenge is:

```
realm="somerealm",nonce="OA6MG9tEQGm2hh",\
qop="auth",charset=utf-8,algorithm=md5-sess
```

Step 5 (alt): Server2 returns error to Server1:

```
<failure xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
  <incorrect-encoding/>
</failure>
</stream:stream>
```

Step 6: Server1 sends a **[BASE64]** encoded response to the challenge:

```
<response xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
dXNlcm5hbWU9ImV4YW1wbGUub3JnIixyZWFsbT0ic29tZXJlYWxtIixub25j
ZT0iT0E2TUc5dEVRR20yaGgiLGNub25jZT0iT0E2TUhYaDZWcVRyUmsiLG5j
PTAwMDAwMDAxLHFvcD1hdXRoLGRpZ2VzdC11cmk9InhtcHAvZXhhbXBsZS5v
cmciLHJlc3BvbnNlPWQzODhkYWQ5MGQ0YmJkNzYwYTE1MjMyMWYyMTQzYWY3
LGNoYXJzZXQ9dXRmLTgK
</response>
```

The decoded response is:

```
username="example.org",realm="somerealm",\
nonce="OA6MG9tEQGm2hh",cnonce="OA6MHXh6VqTrRk",\
nc=00000001,qop=auth,digest-uri="xmpp/example.org",\
response=d388dad90d4bbd760a152321f2143af7,charset=utf-8
```

Step 7: Server2 sends another **[BASE64]** encoded challenge to Server1:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 458
Keysight, EX-1045
Page 458

```
<challenge xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
cnNwYXV0aD1lYTQwZjYwMzM1YzQyN2I1NTI3Yjg0ZGJhYmNkZmZmZAo=
</challenge>
```

The decoded challenge is:

```
rspauth=ea40f60335c427b5527b84dbabcdfffd
```

Step 7 (alt): Server2 returns error to Server1:

```
<failure xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
  <invalid-authzid/>
</failure>
</stream:stream>
```

Step 8: Server1 responds to the challenge:

```
<response xmlns='urn:ietf:params:xml:ns:xmpp-sasl'/>
```

Step 8 (alt): Server1 aborts negotiation:

```
<abort xmlns='urn:ietf:params:xml:ns:xmpp-sasl'/>
```

Step 9: Server2 informs Server1 of successful authentication:

```
<success xmlns='urn:ietf:params:xml:ns:xmpp-sasl'/>
```

Step 9 (alt): Server2 informs Server1 of failed authentication:

```
<failure xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
  <aborted/>
</failure>
</stream:stream>
```

Step 10: Server1 initiates a new stream to Server2:

```
<stream:stream
    xmlns='jabber:server'
    xmlns:stream='http://etherx.jabber.org/streams'
    to='example.com'
    version='1.0'>
```

Step 11: Server2 responds by sending a stream header to Server1 along with any



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 459
Keysight, EX-1045
Page 459

additional features (or an empty features element):

```
<stream:stream
    xmlns='jabber:server'
    xmlns:stream='http://etherx.jabber.org/streams'
    from='example.com'
    id='s2s_345'
    version='1.0'>
<stream:features/>
```

---

## 7. Resource Binding

TOC

After **SASL negotiation** with the receiving entity, the initiating entity MAY want or need to bind a specific resource to that stream. In general this applies only to clients: in order to conform to the **addressing format** and **stanza delivery rules** specified herein, there MUST be a resource identifier associated with the <node@domain> of the client (which is either generated by the server or provided by the client application); this ensures that the address for use over that stream is a "full JID" of the form <node@domain/resource>.

Upon receiving a success indication within the SASL negotiation, the client MUST send a new stream header to the server, to which the server MUST respond with a stream header as well as a list of available stream features. Specifically, if the server requires the client to bind a resource to the stream after successful SASL negotiation, it MUST include an empty <bind/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-bind' namespace in the stream features list it presents to the client upon sending the header for the response stream sent after successful SASL negotiation (but not before):

Server advertises resource binding feature to client:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    id='c2s_345'
    from='example.com'
    version='1.0'>
<stream:features>
  <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'/>
</stream:features>
```

Upon being so informed that resource binding is required, the client MUST bind a resource to the stream by sending to the server an IQ stanza of type "set" (see **IQ Semantics**) containing data qualified by the 'urn:ietf:params:xml:ns:xmpp-bind' namespace.

If the client wishes to allow the server to generate the resource identifier on its behalf, it sends an IQ stanza of type "set" that contains an empty <bind/> element:

Client asks server to bind a resource:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 460
Keysight, EX-1045
Page 460

```
<iq type='set' id='bind_1'>
  <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'/>
</iq>
```

A server that supports resource binding MUST be able to generate a resource identifier on behalf of a client. A resource identifier generated by the server MUST be unique for that <node@domain>.

If the client wishes to specify the resource identifier, it sends an IQ stanza of type "set" that contains the desired resource identifier as the XML character data of a <resource/> element that is a child of the <bind/> element:

Client binds a resource:

```
<iq type='set' id='bind_2'>
  <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'>
    <resource>someresource</resource>
  </bind>
</iq>
```

Once the server has generated a resource identifier for the client or accepted the resource identifier provided by the client, it MUST return an IQ stanza of type "result" to the client, which MUST include a <jid/> child element that specifies the full JID for the connected resource as determined by the server:

Server informs client of successful resource binding:

```
<iq type='result' id='bind_2'>
  <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'>
    <jid>somenode@example.com/someresource</jid>
  </bind>
</iq>
```

A server SHOULD accept the resource identifier provided by the client, but MAY override it with a resource identifier that the server generates; in this case, the server SHOULD NOT return a stanza error (e.g., <forbidden/>) to the client but instead SHOULD communicate the generated resource identifier to the client in the IQ result as shown above.

When a client supplies a resource identifier, the following stanza error conditions are possible (see **Stanza Errors**):

- The provided resource identifier cannot be processed by the server in accordance with **Resourceprep**.
- The client is not allowed to bind a resource to the stream (e.g., because the node or user has reached a limit on the number of connected resources allowed).
- The provided resource identifier is already in use but the server does not allow binding of multiple connected resources with the same identifier.

The protocol for these error conditions is shown below.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 461
Keysight, EX-1045
Page 461

Resource identifier cannot be processed:

```
<iq type='error' id='bind_2'>
  <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'>
    <resource>someresource</resource>
  </bind>
  <error type='modify'>
    <bad-request xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

Client is not allowed to bind a resource:

```
<iq type='error' id='bind_2'>
  <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'>
    <resource>someresource</resource>
  </bind>
  <error type='cancel'>
    <not-allowed xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

Resource identifier is in use:

```
<iq type='error' id='bind_2'>
  <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'>
    <resource>someresource</resource>
  </bind>
  <error type='cancel'>
    <conflict xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

If, before completing the resource binding step, the client attempts to send an XML stanza other than an IQ stanza with a <bind/> child qualified by the 'urn:ietf:params:xml:ns:xmpp-bind' namespace, the server MUST NOT process the stanza and SHOULD return a <not-authorized/> stanza error to the client.

## 8. Server Dialback

TOC

### 8.1. Overview

TOC

The Jabber protocols from which XMPP was adapted include a "server dialback" method for protecting against domain spoofing, thus making it more difficult to spoof XML stanzas. Server dialback is not a security mechanism, and results in weak verification of server identities only (see **Server-to-Server Communications** regarding this method's



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 462
Keysight, EX-1045
Page 462

security characteristics). Domains requiring robust security SHOULD use TLS and SASL; see **Server-to-Server Communications** for details. If SASL is used for server-to-server authentication, dialback SHOULD NOT be used since it is unnecessary. Documentation of dialback is included mainly for the sake of backward-compatibility with existing implementations and deployments.

The server dialback method is made possible by the existence of the Domain Name System (DNS), since one server can (normally) discover the authoritative server for a given domain. Because dialback depends on DNS, inter-domain communications MUST NOT proceed until the Domain Name System (DNS) hostnames asserted by the servers have been resolved (see **Server-to-Server Communications**).

Server dialback is uni-directional, and results in (weak) verification of identities for one stream in one direction. Because server dialback is not an authentication mechanism, mutual authentication is not possible via dialback. Therefore, server dialback MUST be completed in each direction in order to enable bi-directional communications between two domains.

The method for generating and verifying the keys used in server dialback MUST take into account the hostnames being used, the stream ID generated by the receiving server, and a secret known by the authoritative server's network. The stream ID is security-critical in server dialback and therefore MUST be both unpredictable and non-repeating (see **[RANDOM]** for recommendations regarding randomness for security purposes).

Any error that occurs during dialback negotiation MUST be considered a stream error, resulting in termination of the stream and of the underlying TCP connection. The possible error conditions are specified in the protocol description below.

The following terminology applies:

- Originating Server -- the server that is attempting to establish a connection between two domains.
- Receiving Server -- the server that is trying to authenticate that the Originating Server represents the domain which it claims to be.
- Authoritative Server -- the server that answers to the DNS hostname asserted by the Originating Server; for basic environments this will be the Originating Server, but it could be a separate machine in the Originating Server's network.

## 8.2. Order of Events

TOC

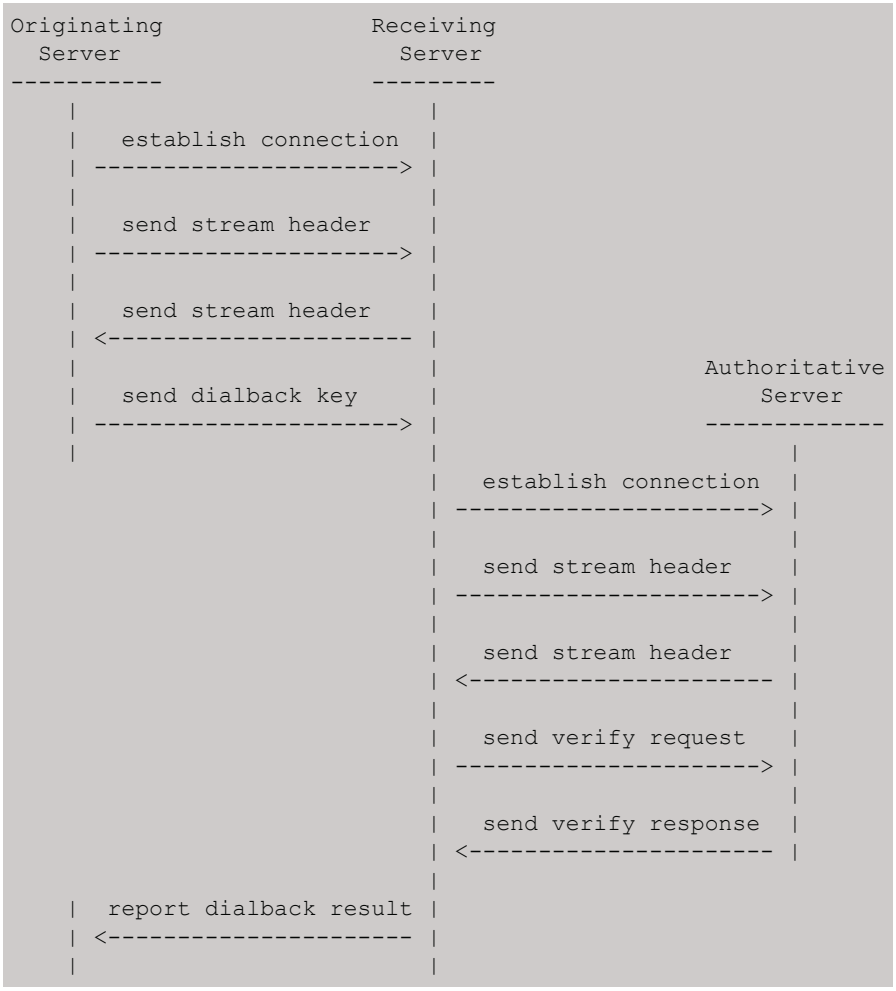
The following is a brief summary of the order of events in dialback:

1. The Originating Server establishes a connection to the Receiving Server.
2. The Originating Server sends a 'key' value over the connection to the Receiving Server.
3. The Receiving Server establishes a connection to the Authoritative Server.
4. The Receiving Server sends the same 'key' value to the Authoritative Server.
5. The Authoritative Server replies that key is valid or invalid.
6. The Receiving Server informs the Originating Server whether it is authenticated or not.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 463
Keysight, EX-1045
Page 463

We can represent this flow of events graphically as follows:




## 8.3. Protocol

TOC

The detailed protocol interaction between the servers is as follows:

1. The Originating Server establishes TCP connection to the Receiving Server.
2. The Originating Server sends a stream header to the Receiving Server:

```
<stream:stream
    xmlns:stream='http://etherx.jabber.org/streams'
    xmlns='jabber:server'
    xmlns:db='jabber:server:dialback'>
```

Note: The 'to' and 'from' attributes are OPTIONAL on the root stream element. The inclusion of the xmlns:db namespace declaration with the name shown indicates to the Receiving Server that the Originating Server supports dialback. If the namespace



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 464
Keysight, EX-1045
Page 464

name is incorrect, then the Receiving Server MUST generate an <invalid-namespace/> stream error condition and terminate both the XML stream and the underlying TCP connection.

3. The Receiving Server SHOULD send a stream header back to the Originating Server, including a unique ID for this interaction:

```
<stream:stream
    xmlns:stream='http://etherx.jabber.org/streams'
    xmlns='jabber:server'
    xmlns:db='jabber:server:dialback'
    id='457F9224A0...'>
```

Note: The 'to' and 'from' attributes are OPTIONAL on the root stream element. If the namespace name is incorrect, then the Originating Server MUST generate an <invalid-namespace/> stream error condition and terminate both the XML stream and the underlying TCP connection. Note well that the Receiving Server SHOULD reply but MAY silently terminate the XML stream and underlying TCP connection depending on security policies in place; however, if the Receiving Server desires to proceed, it MUST send a stream header back to the Originating Server.

4. The Originating Server sends a dialback key to the Receiving Server:

```
<db:result
    to='Receiving Server'
    from='Originating Server'>
  98AF014EDC0...
</db:result>
```

Note: This key is not examined by the Receiving Server, since the Receiving Server does not keep information about the Originating Server between sessions. The key generated by the Originating Server MUST be based in part on the value of the ID provided by the Receiving Server in the previous step, and in part on a secret shared by the Originating Server and the Authoritative Server. If the value of the 'to' address does not match a hostname recognized by the Receiving Server, then the Receiving Server MUST generate a <host-unknown/> stream error condition and terminate both the XML stream and the underlying TCP connection. If the value of the 'from' address matches a domain with which the Receiving Server already has an established connection, then the Receiving Server MUST maintain the existing connection until it validates whether the new connection is legitimate; additionally, the Receiving Server MAY choose to generate a <not-authorized/> stream error condition for the new connection and then terminate both the XML stream and the underlying TCP connection related to the new request.

5. The Receiving Server establishes a TCP connection back to the domain name asserted by the Originating Server, as a result of which it connects to the Authoritative Server. (Note: As an optimization, an implementation MAY reuse an existing connection here.)



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 465
Keysight, EX-1045
Page 465

6. The Receiving Server sends the Authoritative Server a stream header:

```
<stream:stream
    xmlns:stream='http://etherx.jabber.org/streams'
    xmlns='jabber:server'
    xmlns:db='jabber:server:dialback'>
```

Note: The 'to' and 'from' attributes are OPTIONAL on the root stream element. If the namespace name is incorrect, then the Authoritative Server MUST generate an <invalid-namespace/> stream error condition and terminate both the XML stream and the underlying TCP connection.

7. The Authoritative Server sends the Receiving Server a stream header:

```
<stream:stream
    xmlns:stream='http://etherx.jabber.org/streams'
    xmlns='jabber:server'
    xmlns:db='jabber:server:dialback'
    id='1251A342B...'>
```

Note: If the namespace name is incorrect, then the Receiving Server MUST generate an <invalid-namespace/> stream error condition and terminate both the XML stream and the underlying TCP connection between it and the Authoritative Server. If a stream error occurs between the Receiving Server and the Authoritative Server, then the Receiving Server MUST generate a <remote-connection-failed/> stream error condition and terminate both the XML stream and the underlying TCP connection between it and the Originating Server.

8. The Receiving Server sends the Authoritative Server a request for verification of a key:

```
<db:verify
    from='Receiving Server'
    to='Originating Server'
    id='457F9224A0...'>
  98AF014EDC0...
</db:verify>
```

Note: Passed here are the hostnames, the original identifier from the Receiving Server's stream header to the Originating Server in Step 3, and the key that the Originating Server sent to the Receiving Server in Step 4. Based on this information, as well as shared secret information within the Authoritative Server's network, the key is verified. Any verifiable method MAY be used to generate the key. If the value of the 'to' address does not match a hostname recognized by the Authoritative Server, then the Authoritative Server MUST generate a <host-unknown/> stream error condition and terminate both the XML stream and the underlying TCP connection. If the value of the 'from' address does not match the hostname represented by the Receiving Server when opening the



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 466
Keysight, EX-1045
Page 466

TCP connection (or any validated domain thereof, such as a validated subdomain of the Receiving Server's hostname or another validated domain hosted by the Receiving Server), then the Authoritative Server MUST generate an <invalid-from/> stream error condition and terminate both the XML stream and the underlying TCP connection.

9. The Authoritative Server verifies whether the key was valid or invalid:

```
<db:verify
    from='Originating Server'
    to='Receiving Server'
    type='valid'
    id='457F9224A0...'/>
```

or

```
<db:verify
    from='Originating Server'
    to='Receiving Server'
    type='invalid'
    id='457F9224A0...'/>
```

Note: If the ID does not match that provided by the Receiving Server in Step 3, then the Receiving Server MUST generate an <invalid-id/> stream error condition and terminate both the XML stream and the underlying TCP connection. If the value of the 'to' address does not match a hostname recognized by the Receiving Server, then the Receiving Server MUST generate a <host-unknown/> stream error condition and terminate both the XML stream and the underlying TCP connection. If the value of the 'from' address does not match the hostname represented by the Originating Server when opening the TCP connection (or any validated domain thereof, such as a validated subdomain of the Originating Server's hostname or another validated domain hosted by the Originating Server), then the Receiving Server MUST generate an <invalid-from/> stream error condition and terminate both the XML stream and the underlying TCP connection. After returning the verification to the Receiving Server, the Authoritative Server SHOULD terminate the stream between them.

10. The Receiving Server informs the Originating Server of the result:

```
<db:result
    from='Receiving Server'
    to='Originating Server'
    type='valid'/>
```

Note: At this point, the connection has either been validated via a type='valid', or reported as invalid. If the connection is invalid, then the Receiving Server MUST terminate both the XML stream and the underlying TCP connection. If the connection is validated, data can be sent by the Originating Server and read by the Receiving Server;



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 467
Keysight, EX-1045
Page 467

before that, all XML stanzas sent to the Receiving Server SHOULD be silently dropped.

The result of the foregoing is that the Receiving Server has verified the identity of the Originating Server, so that the Originating Server can send, and the Receiving Server can accept, XML stanzas over the "initial stream" (i.e., the stream from the Originating Server to the Receiving Server). In order to verify the identities of the entities using the "response stream" (i.e., the stream from the Receiving Server to the Originating Server), dialback MUST be completed in the opposite direction as well.

After successful dialback negotiation, the Receiving Server SHOULD accept subsequent <db:result/> packets (e.g., validation requests sent to a subdomain or other hostname serviced by the Receiving Server) from the Originating Server over the existing validated connection; this enables "piggybacking" of the original validated connection in one direction.

Even if dialback negotiation is successful, a server MUST verify that all XML stanzas received from the other server include a 'from' attribute and a 'to' attribute; if a stanza does not meet this restriction, the server that receives the stanza MUST generate an <improper-addressing/> stream error condition and terminate both the XML stream and the underlying TCP connection. Furthermore, a server MUST verify that the 'from' attribute of stanzas received from the other server includes a validated domain for the stream; if a stanza does not meet this restriction, the server that receives the stanza MUST generate an <invalid-from/> stream error condition and terminate both the XML stream and the underlying TCP connection. Both of these checks help to prevent spoofing related to particular stanzas.

## 9. XML Stanzas

TOC

After **TLS negotiation** if desired, **SASL negotiation**, and **Resource Binding** if necessary, XML stanzas can be sent over the streams. Three kinds of XML stanza are defined for the 'jabber:client' and 'jabber:server' namespaces: <message/>, <presence/>, and <iq/>. In addition, there are five common attributes for these kinds of stanza. These common attributes, as well as the basic semantics of the three stanza kinds, are defined herein; more detailed information regarding the syntax of XML stanzas in relation to instant messaging and presence applications is provided in **[XMPP-IM]**.

### 9.1. Common Attributes

TOC

The following five attributes are common to message, presence, and IQ stanzas:

#### 9.1.1. to

TOC

The 'to' attribute specifies the JID of the intended recipient for the stanza.

In the 'jabber:client' namespace, a stanza SHOULD possess a 'to' attribute, although a stanza sent from a client to a server for handling by that server (e.g., presence sent to



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 468
Keysight, EX-1045
Page 468

the server for broadcasting to other entities) SHOULD NOT possess a 'to' attribute.

In the 'jabber:server' namespace, a stanza MUST possess a 'to' attribute; if a server receives a stanza that does not meet this restriction, it MUST generate an <improper-addressing/> stream error condition and terminate both the XML stream and the underlying TCP connection with the offending server.

If the value of the 'to' attribute is invalid or cannot be contacted, the entity discovering that fact (usually the sender's or recipient's server) MUST return an appropriate error to the sender, setting the 'from' attribute of the error stanza to the value provided in the 'to' attribute of the offending stanza.

---

### 9.1.2. from

TOC

The 'from' attribute specifies the JID of the sender.

When a server receives an XML stanza within the context of an authenticated stream qualified by the 'jabber:client' namespace, it MUST do one of the following:

1. validate that the value of the 'from' attribute provided by the client is that of a connected resource for the associated entity
2. add a 'from' address to the stanza whose value is the bare JID (<node@domain>) or the full JID (<node@domain/resource>) determined by the server for the connected resource that generated the stanza (see **Determination of Addresses**)

If a client attempts to send an XML stanza for which the value of the 'from' attribute does not match one of the connected resources for that entity, the server SHOULD return an <invalid-from/> stream error to the client. If a client attempts to send an XML stanza over a stream that is not yet authenticated, the server SHOULD return a <not-authorized/> stream error to the client. If generated, both of these conditions MUST result in closure of the stream and termination of the underlying TCP connection; this helps to prevent a denial of service attack launched from a rogue client.

When a server generates a stanza from the server itself for delivery to a connected client (e.g., in the context of data storage services provided by the server on behalf of the client), the stanza MUST either (1) not include a 'from' attribute or (2) include a 'from' attribute whose value is the account's bare JID (<node@domain>) or client's full JID (<node@domain/resource>). A server MUST NOT send to the client a stanza without a 'from' attribute if the stanza was not generated by the server itself. When a client receives a stanza that does not include a 'from' attribute, it MUST assume that the stanza is from the server to which the client is connected.

In the 'jabber:server' namespace, a stanza MUST possess a 'from' attribute; if a server receives a stanza that does not meet this restriction, it MUST generate an <improper-addressing/> stream error condition. Furthermore, the domain identifier portion of the JID contained in the 'from' attribute MUST match the hostname of the sending server (or any validated domain thereof, such as a validated subdomain of the sending server's hostname or another validated domain hosted by the sending server) as communicated in the SASL negotiation or dialback negotiation; if a server receives a stanza that does not meet this restriction, it MUST generate an <invalid-from/> stream error condition. Both of these conditions MUST result in closure of the stream and termination of the underlying TCP connection; this helps to prevent a denial of service attack launched



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 469
Keysight, EX-1045
Page 469

from a rogue server.

### 9.1.3. id

TOC

The optional 'id' attribute MAY be used by a sending entity for internal tracking of stanzas that it sends and receives (especially for tracking the request-response interaction inherent in the semantics of IQ stanzas). It is OPTIONAL for the value of the 'id' attribute to be unique globally, within a domain, or within a stream. The semantics of IQ stanzas impose additional restrictions; see **IQ Semantics**.

### 9.1.4. type

TOC

The 'type' attribute specifies detailed information about the purpose or context of the message, presence, or IQ stanza. The particular allowable values for the 'type' attribute vary depending on whether the stanza is a message, presence, or IQ; the values for message and presence stanzas are specific to instant messaging and presence applications and therefore are defined in **[XMPP-IM]**, whereas the values for IQ stanzas specify the role of an IQ stanza in a structured request-response "conversation" and thus are defined under **IQ Semantics** below. The only 'type' value common to all three stanzas is "error"; see **Stanza Errors**.

### 9.1.5. xml:lang

TOC

A stanza SHOULD possess an 'xml:lang' attribute (as defined in Section 2.12 of **[XML]**) if the stanza contains XML character data that is intended to be presented to a human user (as explained in **RFC 2277** [CHARSET], "internationalization is for humans"). The value of the 'xml:lang' attribute specifies the default language of any such human-readable XML character data, which MAY be overridden by the 'xml:lang' attribute of a specific child element. If a stanza does not possess an 'xml:lang' attribute, an implementation MUST assume that the default language is that specified for the stream as defined under **Stream Attributes** above. The value of the 'xml:lang' attribute MUST be an NMTOKEN and MUST conform to the format defined in **RFC 3066** [LANGTAGS].

## 9.2. Basic Semantics

TOC

### 9.2.1. Message Semantics

TOC

The <message/> stanza kind can be seen as a "push" mechanism whereby one entity pushes information to another entity, similar to the communications that occur in a system such as email. All message stanzas SHOULD possess a 'to' attribute that specifies the intended recipient of the message; upon receiving such a stanza, a server SHOULD route or deliver it to the intended recipient (see **Server Rules for Handling XML Stanzas** for general routing and delivery rules related to XML stanzas).



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 470
Keysight, EX-1045
Page 470

### 9.2.2. Presence Semantics

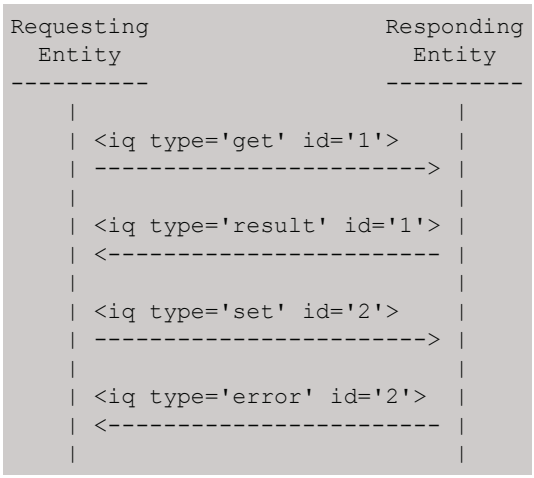
TOC

The <presence/> element can be seen as a basic broadcast or "publish-subscribe" mechanism, whereby multiple entities receive information about an entity to which they have subscribed (in this case, network availability information). In general, a publishing entity SHOULD send a presence stanza with no 'to' attribute, in which case the server to which the entity is connected SHOULD broadcast or multiplex that stanza to all subscribing entities. However, a publishing entity MAY also send a presence stanza with a 'to' attribute, in which case the server SHOULD route or deliver that stanza to the intended recipient. See **Server Rules for Handling XML Stanzas** for general routing and delivery rules related to XML stanzas, and **[XMPP-IM]** for presence-specific rules in the context of an instant messaging and presence application.

### 9.2.3. IQ Semantics

TOC

Info/Query, or IQ, is a request-response mechanism, similar in some ways to **[HTTP]**. The semantics of IQ enable an entity to make a request of, and receive a response from, another entity. The data content of the request and response is defined by the namespace declaration of a direct child element of the IQ element, and the interaction is tracked by the requesting entity through use of the 'id' attribute. Thus, IQ interactions follow a common pattern of structured data exchange such as get/result or set/result (although an error may be returned in reply to a request if appropriate):




In order to enforce these semantics, the following rules apply:

1. The 'id' attribute is REQUIRED for IQ stanzas.
2. The 'type' attribute is REQUIRED for IQ stanzas. The value MUST be one of the following:
    - get -- The stanza is a request for information or requirements.
    - set -- The stanza provides required data, sets new values, or replaces existing values.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 471
Keysight, EX-1045
Page 471

    - result -- The stanza is a response to a successful get or set request.
    - error -- An error has occurred regarding processing or delivery of a previously-sent get or set (see **Stanza Errors**).
3. An entity that receives an IQ request of type "get" or "set" MUST reply with an IQ response of type "result" or "error" (the response MUST preserve the 'id' attribute of the request).
4. An entity that receives a stanza of type "result" or "error" MUST NOT respond to the stanza by sending a further IQ response of type "result" or "error"; however, as shown above, the requesting entity MAY send another request (e.g., an IQ of type "set" in order to provide required information discovered through a get/result pair).
5. An IQ stanza of type "get" or "set" MUST contain one and only one child element that specifies the semantics of the particular request or response.
6. An IQ stanza of type "result" MUST include zero or one child elements.
7. An IQ stanza of type "error" SHOULD include the child element contained in the associated "get" or "set" and MUST include an <error/> child; for details, see **Stanza Errors**.

## 9.3. Stanza Errors

TOC

Stanza-related errors are handled in a manner similar to **stream errors**. However, unlike stream errors, stanza errors are recoverable; therefore error stanzas include hints regarding actions that the original sender can take in order to remedy the error.

### 9.3.1. Rules

TOC

The following rules apply to stanza-related errors:

- The receiving or processing entity that detects an error condition in relation to a stanza MUST return to the sending entity a stanza of the same kind (message, presence, or IQ), whose 'type' attribute is set to a value of "error" (such a stanza is called an "error stanza" herein).
- The entity that generates an error stanza SHOULD include the original XML sent so that the sender can inspect and, if necessary, correct the XML before attempting to resend.
- An error stanza MUST contain an <error/> child element.
- An <error/> child MUST NOT be included if the 'type' attribute has a value other than "error" (or if there is no 'type' attribute).
- An entity that receives an error stanza MUST NOT respond to the stanza with a further error stanza; this helps to prevent looping.

### 9.3.2. Syntax

TOC

The syntax for stanza-related errors is as follows:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 472
Keysight, EX-1045
Page 472

```
<stanza-kind to='sender' type='error'>
  [RECOMMENDED to include sender XML here]
  <error type='error-type'>
    <defined-condition xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <text xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'
          xml:lang='langcode'>
      OPTIONAL descriptive text
    </text>
    [OPTIONAL application-specific condition element]
  </error>
</stanza-kind>
```

The stanza-kind is one of message, presence, or iq.

The value of the <error/> element's 'type' attribute MUST be one of the following:

- cancel -- do not retry (the error is unrecoverable)
- continue -- proceed (the condition was only a warning)
- modify -- retry after changing the data sent
- auth -- retry after providing credentials
- wait -- retry after waiting (the error is temporary)

The <error/> element:

- MUST contain a child element corresponding to one of the defined stanza error conditions specified below; this element MUST be qualified by the 'urn:ietf:params:xml:ns:xmpp-stanzas' namespace.
- MAY contain a <text/> child containing XML character data that describes the error in more detail; this element MUST be qualified by the 'urn:ietf:params:xml:ns:xmpp-stanzas' namespace and SHOULD possess an 'xml:lang' attribute.
- MAY contain a child element for an application-specific error condition; this element MUST be qualified by an application-defined namespace, and its structure is defined by that namespace.

The <text/> element is OPTIONAL. If included, it SHOULD be used only to provide descriptive or diagnostic information that supplements the meaning of a defined condition or application-specific condition. It SHOULD NOT be interpreted programmatically by an application. It SHOULD NOT be used as the error message presented to a user, but MAY be shown in addition to the error message associated with the included condition element (or elements).

Finally, to maintain backward compatibility, the schema (specified in **[XMPP-IM]**) allows the optional inclusion of a 'code' attribute on the <error/> element.

### 9.3.3. Defined Conditions

TOC

The following conditions are defined for use in stanza errors.

- <bad-request/> -- the sender has sent XML that is malformed or that cannot be processed (e.g., an IQ stanza that includes an unrecognized value of the 'type' attribute); the associated error type SHOULD be "modify".



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 473
Keysight, EX-1045
Page 473

- <conflict/> -- access cannot be granted because an existing resource or session exists with the same name or address; the associated error type SHOULD be "cancel".
- <feature-not-implemented/> -- the feature requested is not implemented by the recipient or server and therefore cannot be processed; the associated error type SHOULD be "cancel".
- <forbidden/> -- the requesting entity does not possess the required permissions to perform the action; the associated error type SHOULD be "auth".
- <gone/> -- the recipient or server can no longer be contacted at this address (the error stanza MAY contain a new address in the XML character data of the <gone/> element); the associated error type SHOULD be "modify".
- <internal-server-error/> -- the server could not process the stanza because of a misconfiguration or an otherwise-undefined internal server error; the associated error type SHOULD be "wait".
- <item-not-found/> -- the addressed JID or item requested cannot be found; the associated error type SHOULD be "cancel".
- <jid-malformed/> -- the sending entity has provided or communicated an XMPP address (e.g., a value of the 'to' attribute) or aspect thereof (e.g., a resource identifier) that does not adhere to the syntax defined in **Addressing Scheme**; the associated error type SHOULD be "modify".
- <not-acceptable/> -- the recipient or server understands the request but is refusing to process it because it does not meet criteria defined by the recipient or server (e.g., a local policy regarding acceptable words in messages); the associated error type SHOULD be "modify".
- <not-allowed/> -- the recipient or server does not allow any entity to perform the action; the associated error type SHOULD be "cancel".
- <not-authorized/> -- the sender must provide proper credentials before being allowed to perform the action, or has provided improper credentials; the associated error type SHOULD be "auth".
- <payment-required/> -- the requesting entity is not authorized to access the requested service because payment is required; the associated error type SHOULD be "auth".
- <recipient-unavailable/> -- the intended recipient is temporarily unavailable; the associated error type SHOULD be "wait" (note: an application MUST NOT return this error if doing so would provide information about the intended recipient's network availability to an entity that is not authorized to know such information).
- <redirect/> -- the recipient or server is redirecting requests for this information to another entity, usually temporarily (the error stanza SHOULD contain the alternate address, which MUST be a valid JID, in the XML character data of the <redirect/> element); the associated error type SHOULD be "modify".
- <registration-required/> -- the requesting entity is not authorized to access the requested service because registration is required; the associated error type SHOULD be "auth".
- <remote-server-not-found/> -- a remote server or service specified as part or all of the JID of the intended recipient does not exist; the associated error type SHOULD be "cancel".
- <remote-server-timeout/> -- a remote server or service specified as part or all of the JID of the intended recipient (or required to fulfill a request) could not be contacted within a reasonable amount of time; the associated error



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 474
Keysight, EX-1045
Page 474

type SHOULD be "wait".

- <resource-constraint/> -- the server or recipient lacks the system resources necessary to service the request; the associated error type SHOULD be "wait".
- <service-unavailable/> -- the server or recipient does not currently provide the requested service; the associated error type SHOULD be "cancel".
- <subscription-required/> -- the requesting entity is not authorized to access the requested service because a subscription is required; the associated error type SHOULD be "auth".
- <undefined-condition/> -- the error condition is not one of those defined by the other conditions in this list; any error type may be associated with this condition, and it SHOULD be used only in conjunction with an application-specific condition.
- <unexpected-request/> -- the recipient or server understood the request but was not expecting it at this time (e.g., the request was out of order); the associated error type SHOULD be "wait".

### 9.3.4. Application-Specific Conditions

TOC

As noted, an application MAY provide application-specific stanza error information by including a properly-namespaced child in the error element. The application-specific element SHOULD supplement or further qualify a defined element. Thus, the <error/> element will contain two or three child elements:

```
<iq type='error' id='some-id'>
  <error type='modify'>
    <bad-request xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <too-many-parameters xmlns='application-ns'/>
  </error>
</iq>
```

```
<message type='error' id='another-id'>
  <error type='modify'>
    <undefined-condition
          xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <text xml:lang='en'
          xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'>
      Some special application diagnostic information...
    </text>
    <special-application-condition xmlns='application-ns'/>
  </error>
</message>
```

## 10. Server Rules for Handling XML Stanzas

TOC

Compliant server implementations MUST ensure in-order processing of XML stanzas between any two entities.

Beyond the requirement for in-order processing, each server implementation will contain



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 475
Keysight, EX-1045
Page 475

its own "delivery tree" for handling stanzas it receives. Such a tree determines whether a stanza needs to be routed to another domain, processed internally, or delivered to a resource associated with a connected node. The following rules apply:

## 10.1. No 'to' Address

TOC

If the stanza possesses no 'to' attribute, the server SHOULD process it on behalf of the entity that sent it. Because all stanzas received from other servers MUST possess a 'to' attribute, this rule applies only to stanzas received from a registered entity (such as a client) that is connected to the server. If the server receives a presence stanza with no 'to' attribute, the server SHOULD broadcast it to the entities that are subscribed to the sending entity's presence, if applicable (the semantics of presence broadcast for instant messaging and presence applications are defined in **[XMPP-IM]**). If the server receives an IQ stanza of type "get" or "set" with no 'to' attribute and it understands the namespace that qualifies the content of the stanza, it MUST either process the stanza on behalf of sending entity (where the meaning of "process" is determined by the semantics of the qualifying namespace) or return an error to the sending entity.

## 10.2. Foreign Domain

TOC

If the hostname of the domain identifier portion of the JID contained in the 'to' attribute does not match one of the configured hostnames of the server itself or a subdomain thereof, the server SHOULD route the stanza to the foreign domain (subject to local service provisioning and security policies regarding inter-domain communication). There are two possible cases:

A server-to-server stream already exists between the two domains:
: The sender's server routes the stanza to the authoritative server for the foreign domain over the existing stream

There exists no server-to-server stream between the two domains:
: The sender's server (1) resolves the hostname of the foreign domain (as defined under **Server-to-Server Communications**), (2) negotiates a server-to-server stream between the two domains (as defined under **Use of TLS** and **Use of SASL**), and (3) routes the stanza to the authoritative server for the foreign domain over the newly-established stream

If routing to the recipient's server is unsuccessful, the sender's server MUST return an error to the sender; if the recipient's server can be contacted but delivery by the recipient's server to the recipient is unsuccessful, the recipient's server MUST return an error to the sender by way of the sender's server.

## 10.3. Subdomain

TOC

If the hostname of the domain identifier portion of the JID contained in the 'to' attribute matches a subdomain of one of the configured hostnames of the server itself, the server MUST either process the stanza itself or route the stanza to a specialized service that is responsible for that subdomain (if the subdomain is configured), or return an error to



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 476
Keysight, EX-1045
Page 476

the sender (if the subdomain is not configured).

### 10.4. Mere Domain or Specific Resource

TOC

If the hostname of the domain identifier portion of the JID contained in the 'to' attribute matches a configured hostname of the server itself and the JID contained in the 'to' attribute is of the form <domain> or <domain/resource>, the server (or a defined resource thereof) MUST either process the stanza as appropriate for the stanza kind or return an error stanza to the sender.

### 10.5. Node in Same Domain

TOC

If the hostname of the domain identifier portion of the JID contained in the 'to' attribute matches a configured hostname of the server itself and the JID contained in the 'to' attribute is of the form <node@domain> or <node@domain/resource>, the server SHOULD deliver the stanza to the intended recipient of the stanza as represented by the JID contained in the 'to' attribute. The following rules apply:

1. If the JID contains a resource identifier (i.e., is of the form <node@domain/resource>) and there exists a connected resource that matches the full JID, the recipient's server SHOULD deliver the stanza to the stream or session that exactly matches the resource identifier.
2. If the JID contains a resource identifier and there exists no connected resource that matches the full JID, the recipient's server SHOULD return a <service-unavailable/> stanza error to the sender.
3. If the JID is of the form <node@domain> and there exists at least one connected resource for the node, the recipient's server SHOULD deliver the stanza to at least one of the connected resources, according to application-specific rules (a set of delivery rules for instant messaging and presence applications is defined in **[XMPP-IM]**).

## 11. XML Usage within XMPP

TOC

### 11.1. Restrictions

TOC

XMPP is a simplified and specialized protocol for streaming XML elements in order to exchange structured information in close to real time. Because XMPP does not require the parsing of arbitrary and complete XML documents, there is no requirement that XMPP needs to support the full feature set of **[XML]**. In particular, the following restrictions apply.

With regard to XML generation, an XMPP implementation MUST NOT inject into an XML stream any of the following:

- comments (as defined in Section 2.5 of **[XML]**)



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 477
Keysight, EX-1045
Page 477

- processing instructions (Section 2.6 therein)
- internal or external DTD subsets (Section 2.8 therein)
- internal or external entity references (Section 4.2 therein) with the exception of predefined entities (Section 4.6 therein)
- character data or attribute values containing unescaped characters that map to the predefined entities (Section 4.6 therein); such characters MUST be escaped

With regard to XML processing, if an XMPP implementation receives such restricted XML data, it MUST ignore the data.

## 11.2. XML Namespace Names and Prefixes

TOC

**XML Namespaces** [XML-NAMES] are used within all XMPP-compliant XML to create strict boundaries of data ownership. The basic function of namespaces is to separate different vocabularies of XML elements that are structurally mixed together. Ensuring that XMPP-compliant XML is namespace-aware enables any allowable XML to be structurally mixed with any data element within XMPP. Rules for XML namespace names and prefixes are defined in the following subsections.

### 11.2.1. Streams Namespace

TOC

A streams namespace declaration is REQUIRED in all XML stream headers. The name of the streams namespace MUST be 'http://etherx.jabber.org/streams'. The element names of the <stream/> element and its <features/> and <error/> children MUST be qualified by the streams namespace prefix in all instances. An implementation SHOULD generate only the 'stream:' prefix for these elements, and for historical reasons MAY accept only the 'stream:' prefix.

### 11.2.2. Default Namespace

TOC

A default namespace declaration is REQUIRED and is used in all XML streams in order to define the allowable first-level children of the root stream element. This namespace declaration MUST be the same for the initial stream and the response stream so that both streams are qualified consistently. The default namespace declaration applies to the stream and all stanzas sent within a stream (unless explicitly qualified by another namespace, or by the prefix of the streams namespace or the dialback namespace).

A server implementation MUST support the following two default namespaces (for historical reasons, some implementations MAY support only these two default namespaces):

- jabber:client -- this default namespace is declared when the stream is used for communications between a client and a server
- jabber:server -- this default namespace is declared when the stream is used for communications between two servers

A client implementation MUST support the 'jabber:client' default namespace, and for



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 478
Keysight, EX-1045
Page 478

historical reasons MAY support only that default namespace.

An implementation MUST NOT generate namespace prefixes for elements in the default namespace if the default namespace is 'jabber:client' or 'jabber:server'. An implementation SHOULD NOT generate namespace prefixes for elements qualified by content (as opposed to stream) namespaces other than 'jabber:client' and 'jabber:server'.

Note: The 'jabber:client' and 'jabber:server' namespaces are nearly identical but are used in different contexts (client-to-server communications for 'jabber:client' and server-to-server communications for 'jabber:server'). The only difference between the two is that the 'to' and 'from' attributes are OPTIONAL on stanzas sent within 'jabber:client', whereas they are REQUIRED on stanzas sent within 'jabber:server'. If a compliant implementation accepts a stream that is qualified by the 'jabber:client' or 'jabber:server' namespace, it MUST support the **common attributes** and **basic semantics** of all three core stanza kinds (message, presence, and IQ).

### 11.2.3. Dialback Namespace

TOC

A dialback namespace declaration is REQUIRED for all elements used in **server dialback**. The name of the dialback namespace MUST be 'jabber:server:dialback'. All elements qualified by this namespace MUST be prefixed. An implementation SHOULD generate only the 'db:' prefix for such elements and MAY accept only the 'db:' prefix.

## 11.3. Validation

TOC

Except as noted with regard to 'to' and 'from' addresses for stanzas within the 'jabber:server' namespace, a server is not responsible for validating the XML elements forwarded to a client or another server; an implementation MAY choose to provide only validated data elements but this is OPTIONAL (although an implementation MUST NOT accept XML that is not well-formed). Clients SHOULD NOT rely on the ability to send data which does not conform to the schemas, and SHOULD ignore any non-conformant elements or attributes on the incoming XML stream. Validation of XML streams and stanzas is OPTIONAL, and schemas are included herein for descriptive purposes only.

## 11.4. Inclusion of Text Declaration

TOC

Implementations SHOULD send a text declaration before sending a stream header. Applications MUST follow the rules in **[XML]** regarding the circumstances under which a text declaration is included.

## 11.5. Character Encoding

TOC

Implementations MUST support the UTF-8 (**RFC 3629** [UTF-8]) transformation of Universal Character Set (**ISO/IEC 10646-1** [UCS2]) characters, as required by **RFC**



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 479
Keysight, EX-1045
Page 479

**2277** [CHARSET]. Implementations MUST NOT attempt to use any other encoding.

## 12. Core Compliance Requirements

TOC

This section summarizes the specific aspects of the Extensible Messaging and Presence Protocol that MUST be supported by servers and clients in order to be considered compliant implementations, as well as additional protocol aspects that SHOULD be supported. For compliance purposes, we draw a distinction between core protocols (which MUST be supported by any server or client, regardless of the specific application) and instant messaging protocols (which MUST be supported only by instant messaging and presence applications built on top of the core protocols). Compliance requirements that apply to all servers and clients are specified in this section; compliance requirements for instant messaging servers and clients are specified in the corresponding section of **[XMPP-IM]**.

### 12.1. Servers

TOC

In addition to all defined requirements with regard to security, XML usage, and internationalization, a server MUST support the following core protocols in order to be considered compliant:

- Application of the **[NAMEPREP]**, **Nodeprep**, and **Resourceprep** profiles of **[STRINGPREP]** to addresses (including ensuring that domain identifiers are internationalized domain names as defined in **[IDNA]**)
- **XML streams**, including **Use of TLS**, **Use of SASL**, and **Resource Binding**
- The basic semantics of the three defined stanza kinds (i.e., <message/>, <presence/>, and <iq/>) as specified in **stanza semantics**
- Generation (and, where appropriate, handling) of error syntax and semantics related to streams, TLS, SASL, and XML stanzas

In addition, a server MAY support the following core protocol:

- **Server dialback**

### 12.2. Clients

TOC

A client MUST support the following core protocols in order to be considered compliant:

- **XML streams**, including **Use of TLS**, **Use of SASL**, and **Resource Binding**
- The basic semantics of the three defined stanza kinds (i.e., <message/>, <presence/>, and <iq/>) as specified in **stanza semantics**
- Handling (and, where appropriate, generation) of error syntax and semantics related to streams, TLS, SASL, and XML stanzas

In addition, a client SHOULD support the following core protocols:

- Generation of addresses to which the **[NAMEPREP]**, **Nodeprep**, and **Resourceprep** profiles of **[STRINGPREP]** can be applied without failing



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 480
Keysight, EX-1045
Page 480

## 13. Internationalization Considerations

TOC

XML streams MUST be encoded in UTF-8 as specified under **Character Encoding**. As specified under **Stream Attributes**, an XML stream SHOULD include an 'xml:lang' attribute that is treated as the default language for any XML character data sent over the stream that is intended to be presented to a human user. As specified under **xml:lang**, an XML stanza SHOULD include an 'xml:lang' attribute if the stanza contains XML character data that is intended to be presented to a human user. A server SHOULD apply the default 'xml:lang' attribute to stanzas it routes or delivers on behalf of connected entities, and MUST NOT modify or delete 'xml:lang' attributes from stanzas it receives from other entities.

## 14. Security Considerations

TOC

### 14.1. High Security

TOC

For the purposes of XMPP communications (client-to-server and server-to-server), the term "high security" refers to the use of security technologies that provide both mutual authentication and integrity-checking; in particular, when using certificate-based authentication to provide high security, a chain-of-trust SHOULD be established out-of-band, although a shared certificate authority signing certificates could allow a previously unknown certificate to establish trust in-band. See **Section 14.2** below regarding certificate validation procedures.

Implementations MUST support high security. Service provisioning SHOULD use high security, subject to local security policies.

### 14.2. Certificate Validation

TOC

When an XMPP peer communicates with another peer securely, it MUST validate the peer's certificate. There are three possible cases:

Case #1:
: The peer contains an End Entity certificate which appears to be certified by a chain of certificates terminating in a trust anchor (as described in Section 6.1 of **[X509]**).

Case #2:
: The peer certificate is certified by a Certificate Authority not known to the validating peer.

Case #3:
: The peer certificate is self-signed.

In Case #1, the validating peer MUST do one of two things:

1. Verify the peer certificate according to the rules of **[X509]**. The certificate



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 481
Keysight, EX-1045
Page 481

SHOULD then be checked against the expected identity of the peer following the rules described in **[HTTP-TLS]**, except that a subjectAltName extension of type "xmpp" MUST be used as the identity if present. If one of these checks fails, user-oriented clients MUST either notify the user (clients MAY give the user the opportunity to continue with the connection in any case) or terminate the connection with a bad certificate error. Automated clients SHOULD terminate the connection (with a bad certificate error) and log the error to an appropriate audit log. Automated clients MAY provide a configuration setting that disables this check, but MUST provide a setting that enables it.

2. The peer SHOULD show the certificate to a user for approval, including the entire certificate chain. The peer MUST cache the certificate (or some non-forgeable representation such as a hash). In future connections, the peer MUST verify that the same certificate was presented and MUST notify the user if it has changed.

In Case #2 and Case #3, implementations SHOULD act as in (2) above.

## 14.3. Client-to-Server Communications

TOC

A compliant client implementation MUST support both TLS and SASL for connections to a server.

The TLS protocol for encrypting XML streams (defined under **Use of TLS**) provides a reliable mechanism for helping to ensure the confidentiality and data integrity of data exchanged between two entities.

The SASL protocol for authenticating XML streams (defined under **Use of SASL**) provides a reliable mechanism for validating that a client connecting to a server is who it claims to be.

Client-to-server communications MUST NOT proceed until the DNS hostname asserted by the server has been resolved. Such resolutions SHOULD first attempt to resolve the hostname using an **[SRV]** Service of "xmpp-client" and Proto of "tcp", resulting in resource records such as "_xmpp-client._tcp.example.com." (the use of the string "xmpp-client" for the service identifier is consistent with the IANA registration). If the SRV lookup fails, the fallback is a normal IPv4/IPv6 address record resolution to determine the IP address, using the "xmpp-client" port 5222, registered with the IANA.

The IP address and method of access of clients MUST NOT be made public by a server, nor are any connections other than the original server connection required. This helps to protect the client's server from direct attack or identification by third parties.

## 14.4. Server-to-Server Communications

TOC

A compliant server implementation MUST support both TLS and SASL for inter-domain communications. For historical reasons, a compliant implementation SHOULD also support **Server Dialback**.

Because service provisioning is a matter of policy, it is OPTIONAL for any given domain to communicate with other domains, and server-to-server communications MAY be

disabled by the administrator of any given deployment. If a particular domain enables inter-domain communications, it SHOULD enable high security.

Administrators may want to require use of SASL for server-to-server communications in order to ensure both authentication and confidentiality (e.g., on an organization's private network). Compliant implementations SHOULD support SASL for this purpose.

Inter-domain connections MUST NOT proceed until the DNS hostnames asserted by the servers have been resolved. Such resolutions MUST first attempt to resolve the hostname using an **[SRV]** Service of "xmpp-server" and Proto of "tcp", resulting in resource records such as "_xmpp-server._tcp.example.com." (the use of the string "xmpp-server" for the service identifier is consistent with the IANA registration; note well that the "xmpp-server" service identifier supersedes the earlier use of a "jabber" service identifier, since the earlier usage did not conform to **[SRV]**; implementations desiring to be backward compatible should continue to look for or answer to the "jabber" service identifier as well). If the SRV lookup fails, the fallback is a normal IPv4/IPv6 address record resolution to determine the IP address, using the "xmpp-server" port 5269, registered with the IANA.

Server dialback helps protect against domain spoofing, thus making it more difficult to spoof XML stanzas. It is not a mechanism for authenticating, securing, or encrypting streams between servers as is done via SASL and TLS, and results in weak verification of server identities only. Furthermore, it is susceptible to DNS poisoning attacks unless **DNSSec** [DNSSEC] is used, and even if the DNS information is accurate, dialback cannot protect from attacks where the attacker is capable of hijacking the IP address of the remote domain. Domains requiring robust security SHOULD use TLS and SASL. If SASL is used for server-to-server authentication, dialback SHOULD NOT be used since it is unnecessary.

## 14.5. Order of Layers

TOC

The order of layers in which protocols MUST be stacked is as follows:

1. TCP
2. TLS
3. SASL
4. XMPP

The rationale for this order is that **[TCP]** is the base connection layer used by all of the protocols stacked on top of TCP, **[TLS]** is often provided at the operating system layer, **[SASL]** is often provided at the application layer, and XMPP is the application itself.

## 14.6. Lack of SASL Channel Binding to TLS

TOC

The SASL framework does not provide a mechanism to bind SASL authentication to a security layer providing confidentiality and integrity protection that was negotiated at a lower layer. This lack of a "channel binding" prevents SASL from being able to verify that the source and destination end points to which the lower layer's security is bound are equivalent to the end points that SASL is authenticating. If the end points are not identical, the lower layer's security cannot be trusted to protect data transmitted



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 483
Keysight, EX-1045
Page 483

between the SASL authenticated entities. In such a situation, a SASL security layer should be negotiated that effectively ignores the presence of the lower layer security.

## 14.7. Mandatory-to-Implement Technologies

TOC

At a minimum, all implementations MUST support the following mechanisms:

for authentication:
: the SASL **[DIGEST-MD5]** mechanism

for confidentiality:
: TLS (using the TLS_RSA_WITH_3DES_EDE_CBC_SHA cipher)

for both:
: TLS plus SASL EXTERNAL(using the TLS_RSA_WITH_3DES_EDE_CBC_SHA cipher supporting client-side certificates)

## 14.8. Firewalls

TOC

Communications using XMPP normally occur over **[TCP]** connections on port 5222 (client-to-server) or port 5269 (server-to-server), as registered with the IANA (see **IANA Considerations**). Use of these well-known ports allows administrators to easily enable or disable XMPP activity through existing and commonly-deployed firewalls.

## 14.9. Use of base64 in SASL

TOC

Both the client and the server MUST verify any **[BASE64]** data received during SASL negotiation. An implementation MUST reject (not ignore) any characters that are not explicitly allowed by the base64 alphabet; this helps to guard against creation of a covert channel that could be used to "leak" information. An implementation MUST NOT break on invalid input and MUST reject any sequence of base64 characters containing the pad ('=') character if that character is included as something other than the last character of the data (e.g., "=AAA" or "BBBB=CCC"); this helps to guard against buffer overflow attacks and other attacks on the implementation. Base 64 encoding visually hides otherwise easily recognized information, such as passwords, but does not provide any computational confidentiality. Base 64 encoding MUST follow the definition in Section 3 of **RFC 3548** [BASE64].

## 14.10. Stringprep Profiles

TOC

XMPP makes use of the **[NAMEPREP]** profile of **[STRINGPREP]** for processing of domain identifiers; for security considerations related to Nameprep, refer to the appropriate section of **[NAMEPREP]**.

In addition, XMPP defines two profiles of **[STRINGPREP]**: **Nodeprep** for node identifiers and **Resourceprep** for resource identifiers.

The Unicode and ISO/IEC 10646 repertoires have many characters that look similar. In



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 484
Keysight, EX-1045
Page 484

many cases, users of security protocols might do visual matching, such as when comparing the names of trusted third parties. Because it is impossible to map similar-looking characters without a great deal of context, such as knowing the fonts used, stringprep does nothing to map similar-looking characters together, nor to prohibit some characters because they look like others.

A node identifier can be employed as one part of an entity's address in XMPP. One common usage is as the username of an instant messaging user; another is as the name of a multi-user chat room; many other kinds of entities could use node identifiers as part of their addresses. The security of such services could be compromised based on different interpretations of the internationalized node identifier; for example, a user entering a single internationalized node identifier could access another user's account information, or a user could gain access to an otherwise restricted chat room or service.

A resource identifier can be employed as one part of an entity's address in XMPP. One common usage is as the name for an instant messaging user's connected resource (active session); another is as the nickname of a user in a multi-user chat room; many other kinds of entities could use resource identifiers as part of their addresses. The security of such services could be compromised based on different interpretations of the internationalized resource identifier; for example, a user could attempt to initiate multiple sessions with the same name, or a user could send a message to someone other than the intended recipient in a multi-user chat room.

## 15. IANA Considerations

TOC

### 15.1. XML Namespace Name for TLS Data

TOC

A URN sub-namespace for TLS-related data in the Extensible Messaging and Presence Protocol (XMPP) is defined as follows. (This namespace name adheres to the format defined in **The IETF XML Registry** [XML-REG].)

URI:
urn:ietf:params:xml:ns:xmpp-tls
Specification:
RFC 3920
Description:
This is the XML namespace name for TLS-related data in the Extensible Messaging and Presence Protocol (XMPP) as defined by RFC 3920.
Registrant Contact:
IETF, XMPP Working Group, <xmppwg@jabber.org>

### 15.2. XML Namespace Name for SASL Data

TOC

A URN sub-namespace for SASL-related data in the Extensible Messaging and Presence Protocol (XMPP) is defined as follows. (This namespace name adheres to the format defined in **[XML-REG]**.)

URI:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 485
Keysight, EX-1045
Page 485

urn:ietf:params:xml:ns:xmpp-sasl
Specification:
RFC 3920
Description:
This is the XML namespace name for SASL-related data in the Extensible Messaging and Presence Protocol (XMPP) as defined by RFC 3920.
Registrant Contact:
IETF, XMPP Working Group, <xmppwg@jabber.org>

### 15.3. XML Namespace Name for Stream Errors

TOC

A URN sub-namespace for stream-related error data in the Extensible Messaging and Presence Protocol (XMPP) is defined as follows. (This namespace name adheres to the format defined in **[XML-REG]**.)

URI:
urn:ietf:params:xml:ns:xmpp-streams
Specification:
RFC 3920
Description:
This is the XML namespace name for stream-related error data in the Extensible Messaging and Presence Protocol (XMPP) as defined by RFC 3920.
Registrant Contact:
IETF, XMPP Working Group, <xmppwg@jabber.org>

### 15.4. XML Namespace Name for Resource Binding

TOC

A URN sub-namespace for resource binding in the Extensible Messaging and Presence Protocol (XMPP) is defined as follows. (This namespace name adheres to the format defined in **[XML-REG]**.)

URI:
urn:ietf:params:xml:ns:xmpp-bind
Specification:
RFC 3920
Description:
This is the XML namespace name for resource binding in the Extensible Messaging and Presence Protocol (XMPP) as defined by RFC 3920.
Registrant Contact:
IETF, XMPP Working Group, <xmppwg@jabber.org>

### 15.5. XML Namespace Name for Stanza Errors

TOC

A URN sub-namespace for stanza-related error data in the Extensible Messaging and Presence Protocol (XMPP) is defined as follows. (This namespace name adheres to the format defined in **[XML-REG]**.)

URI:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 486
Keysight, EX-1045
Page 486

urn:ietf:params:xml:ns:xmpp-stanzas
Specification:
RFC 3920
Description:
This is the XML namespace name for stanza-related error data in the Extensible Messaging and Presence Protocol (XMPP) as defined by RFC 3920.
Registrant Contact:
IETF, XMPP Working Group, <xmppwg@jabber.org>

## 15.6. Nodeprep Profile of Stringprep

TOC

The Nodeprep profile of stringprep is defined under **Nodeprep**. The IANA has registered Nodeprep in the stringprep profile registry.

Name of this profile:

Nodeprep

RFC in which the profile is defined:

RFC 3920

Indicator whether or not this is the newest version of the profile:

This is the first version of Nodeprep

## 15.7. Resourceprep Profile of Stringprep

TOC

The Resourceprep profile of stringprep is defined under **Resourceprep**. The IANA has registered Resourceprep in the stringprep profile registry.

Name of this profile:

Resourceprep

RFC in which the profile is defined:

RFC 3920

Indicator whether or not this is the newest version of the profile:

This is the first version of Resourceprep

## 15.8. GSSAPI Service Name

TOC

The IANA has registered "xmpp" as a **GSSAPI** [GSS-API] service name, as defined under **SASL Definition**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 487
Keysight, EX-1045
Page 487

### 15.9. Port Numbers

TOC

The IANA has registered "xmpp-client" and "xmpp-server" as keywords for **[TCP]** ports 5222 and 5269 respectively.

These ports SHOULD be used for client-to-server and server-to-server communications respectively, but their use is OPTIONAL.

## 16. References

TOC

### 16.1. Normative References

TOC


**[ABNF]** Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF," RFC 2234, November 1997.

**[BASE64]** Josefsson, S., "The Base16, Base32, and Base64 Data Encodings," RFC 3548, July 2003.

**[CHARSET]** Alvestrand, H., "IETF Policy on Character Sets and Languages," BCP 18, RFC 2277, January 1998 (TXT, HTML, XML).

**[DIGEST-MD5]** Leach, P. and C. Newman, "Using Digest Authentication as a SASL Mechanism," RFC 2831, May 2000.

**[DNS]** Mockapetris, P., "Domain names - implementation and specification," STD 13, RFC 1035, November 1987.

**[GSS-API]** Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1," RFC 2743, January 2000.

**[HTTP-TLS]** Rescorla, E., "HTTP Over TLS," RFC 2818, May 2000.

**[IDNA]** Faltstrom, P., Hoffman, P., and A. Costello, "Internationalizing Domain Names in Applications (IDNA)," RFC 3490, March 2003.

**[IPv6]** Hinden, R. and S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture," RFC 3513, April 2003.

**[LANGTAGS]** Alvestrand, H., "Tags for the Identification of Languages," BCP 47, RFC 3066, January 2001.

**[NAMEPREP]** Hoffman, P. and M. Blanchet, "Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN)," RFC 3491, March 2003.

**[RANDOM]** Eastlake, D., Crocker, S., and J. Schiller, "Randomness Recommendations for Security," RFC 1750, December 1994.

**[SASL]** Myers, J., "Simple Authentication and Security Layer (SASL)," RFC 2222, October 1997 (TXT, HTML, XML).

**[SRV]** Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)," RFC 2782, February 2000.

**[STRINGPREP]** Hoffman, P. and M. Blanchet, "Preparation of Internationalized Strings ("stringprep")," RFC 3454, December 2002.

**[TCP]** Postel, J., "Transmission Control Protocol," STD 7, RFC 793, September 1981.

**[TERMS]** Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).

**[TLS]** Dierks, T., Allen, C., Treese, W., Karlton, P., Freier, A., and P. Kocher, "The TLS Protocol Version 1.0," RFC 2246, January 1999.

**[UCS2]** International Organization for Standardization, "Information Technology - Universal Multiple-octet coded Character Set (UCS) - Amendment 2: UCS Transformation Format 8 (UTF-8)," ISO Standard 10646-1 Addendum 2, October 1996.

**[UTF-8]** Yergeau, F., "UTF-8, a transformation format of ISO 10646," STD 63, RFC 3629, November 2003.

**[X509]** Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," RFC 3280, April 2002.

**[XML]** Bray, T., Paoli, J., Sperberg-McQueen, C., and E. Maler, "Extensible Markup Language (XML) 1.0 (2nd ed)," W3C REC-xml, October 2000.

**[XML-NAMES]** Bray, T., Hollander, D., and A. Layman, "Namespaces in XML," W3C REC-xml-names, January 1999.


https://web.archive.org/web/20061010092431/https://xmpp.org/rfcs/rfc3920.html

## 16.2. Informative References


**[ACAP]** Newman, C. and J. Myers, "ACAP -- Application Configuration Access Protocol," RFC 2244, November 1997.

**[ASN.1]** CCITT, "Recommendation X.208: Specification of Abstract Syntax Notation One (ASN.1)," 1988.

**[DNSSEC]** Eastlake, D., "Domain Name System Security Extensions," RFC 2535, March 1999.

**[HTTP]** Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1," RFC 2616, June 1999 (TXT, PS, PDF, HTML, XML).

**[IMAP]** Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1," RFC 3501, March 2003.

**[IMP-REQS]** Day, M., Aggarwal, S., and J. Vincent, "Instant Messaging / Presence Protocol Requirements," RFC 2779, February 2000.

**[IRC]** Oikarinen, J. and D. Reed, "Internet Relay Chat Protocol," RFC 1459, May 1993.

**[JEP-0029]** Kaes, C., "Definition of Jabber Identifiers (JIDs)," JSF JEP 0029, October 2003.

**[JEP-0078]** Saint-Andre, P., "Non-SASL Authentication," JSF JEP 0078, July 2004.

**[JEP-0086]** Norris, R. and P. Saint-Andre, "Error Condition Mappings," JSF JEP 0086, February 2004.

**[JSF]** Jabber Software Foundation, "Jabber Software Foundation."

**[POP3]** Myers, J. and M. Rose, "Post Office Protocol - Version 3," STD 53, RFC 1939, May 1996.

**[SIMPLE]** SIMPLE Working Group, "SIMPLE WG."

**[SMTP]** Klensin, J., "Simple Mail Transfer Protocol," RFC 2821, April 2001.

**[URI]** Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," RFC 2396, August 1998 (TXT, HTML, XML).

**[USINGTLS]** Newman, C., "Using TLS with IMAP, POP3 and ACAP," RFC 2595, June 1999.

**[XML-REG]** Mealling, M., "The IETF XML Registry," BCP 81, RFC 3688, January 2004.

**[XMPP-IM]** Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence," RFC 3921, October 2004.


# Appendix A. Nodeprep

## A.1. Introduction

This appendix defines the "Nodeprep" profile of **[STRINGPREP]**. As such, it specifies processing rules that will enable users to enter internationalized node identifiers in the Extensible Messaging and Presence Protocol (XMPP) and have the highest chance of getting the content of the strings correct. (An XMPP node identifier is the optional portion of an XMPP address that precedes a domain identifier and the '@' separator; it is often but not exclusively associated with an instant messaging username.) These processing rules are intended only for XMPP node identifiers and are not intended for arbitrary text or any other aspect of an XMPP address.

This profile defines the following, as required by **[STRINGPREP]**:

- The intended applicability of the profile: internationalized node identifiers within XMPP
- The character repertoire that is the input and output to stringprep: Unicode 3.2, specified in Section 2 of this Appendix
- The mappings used: specified in Section 3
- The Unicode normalization used: specified in Section 4
- The characters that are prohibited as output: specified in Section 5
- Bidirectional character handling: specified in Section 6



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 489
Keysight, EX-1045
Page 489

## A.2. Character Repertoire

TOC

This profile uses Unicode 3.2 with the list of unassigned code points being Table A.1, both defined in Appendix A of **[STRINGPREP]**.

## A.3. Mapping

TOC

This profile specifies mapping using the following tables from **[STRINGPREP]**:

Table B.1
Table B.2

## A.4. Normalization

TOC

This profile specifies the use of Unicode normalization form KC, as described in **[STRINGPREP]**.

## A.5. Prohibited Output

TOC

This profile specifies the prohibition of using the following tables from **[STRINGPREP]**.

Table C.1.1
Table C.1.2
Table C.2.1
Table C.2.2
Table C.3
Table C.4
Table C.5
Table C.6
Table C.7
Table C.8
Table C.9

In addition, the following Unicode characters are also prohibited:

#x22 (")
#x26 (&)
#x27 (')
#x2F (/)
#x3A (:)
#x3C (<)
#x3E (>)
#x40 (@)



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 490
Keysight, EX-1045
Page 490

### A.6. Bidirectional Characters

TOC

This profile specifies checking bidirectional strings, as described in Section 6 of **[STRINGPREP]**.

## Appendix B. Resourceprep

TOC

### B.1. Introduction

TOC

This appendix defines the "Resourceprep" profile of **[STRINGPREP]**. As such, it specifies processing rules that will enable users to enter internationalized resource identifiers in the Extensible Messaging and Presence Protocol (XMPP) and have the highest chance of getting the content of the strings correct. (An XMPP resource identifier is the optional portion of an XMPP address that follows a domain identifier and the '/' separator; it is often but not exclusively associated with an instant messaging session name.) These processing rules are intended only for XMPP resource identifiers and are not intended for arbitrary text or any other aspect of an XMPP address.

This profile defines the following, as required by **[STRINGPREP]**:

- The intended applicability of the profile: internationalized resource identifiers within XMPP
- The character repertoire that is the input and output to stringprep: Unicode 3.2, specified in Section 2 of this Appendix
- The mappings used: specified in Section 3
- The Unicode normalization used: specified in Section 4
- The characters that are prohibited as output: specified in Section 5
- Bidirectional character handling: specified in Section 6

### B.2. Character Repertoire

TOC

This profile uses Unicode 3.2 with the list of unassigned code points being Table A.1, both defined in Appendix A of **[STRINGPREP]**.

### B.3. Mapping

TOC

This profile specifies mapping using the following tables from **[STRINGPREP]**:

Table B.1

### B.4. Normalization

TOC

This profile specifies the use of Unicode normalization form KC, as described in **[STRINGPREP]**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 491
Keysight, EX-1045
Page 491

### B.5. Prohibited Output

TOC

This profile specifies the prohibition of using the following tables from **[STRINGPREP]**.

Table C.1.2
Table C.2.1
Table C.2.2
Table C.3
Table C.4
Table C.5
Table C.6
Table C.7
Table C.8
Table C.9

### B.6. Bidirectional Characters

TOC

This profile specifies checking bidirectional strings, as described in Section 6 of **[STRINGPREP]**.

## Appendix C. XML Schemas

TOC

The following XML schemas are descriptive, not normative. For schemas defining the 'jabber:client' and 'jabber:server' namespaces, refer to **[XMPP-IM]**.

### C.1. Streams namespace

TOC

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='http://etherx.jabber.org/streams'
    xmlns='http://etherx.jabber.org/streams'
    elementFormDefault='unqualified'>

  <xs:element name='stream'>
    <xs:complexType>
      <xs:sequence xmlns:client='jabber:client'
                   xmlns:server='jabber:server'
                   xmlns:db='jabber:server:dialback'>
        <xs:element ref='features' minOccurs='0' maxOccurs='1'/>
        <xs:any namespace='urn:ietf:params:xml:ns:xmpp-tls'
                minOccurs='0'
                maxOccurs='unbounded'/>
        <xs:any namespace='urn:ietf:params:xml:ns:xmpp-sasl'
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 492
Keysight, EX-1045
Page 492

```
                minOccurs='0'
                maxOccurs='unbounded'/>
        <xs:choice minOccurs='0' maxOccurs='1'>
          <xs:choice minOccurs='0' maxOccurs='unbounded'>
            <xs:element ref='client:message'/>
            <xs:element ref='client:presence'/>
            <xs:element ref='client:iq'/>
          </xs:choice>
          <xs:choice minOccurs='0' maxOccurs='unbounded'>
            <xs:element ref='server:message'/>
            <xs:element ref='server:presence'/>
            <xs:element ref='server:iq'/>
            <xs:element ref='db:result'/>
            <xs:element ref='db:verify'/>
          </xs:choice>
        </xs:choice>
        <xs:element ref='error' minOccurs='0' maxOccurs='1'/>
      </xs:sequence>
      <xs:attribute name='from' type='xs:string' use='optional'/>
      <xs:attribute name='id' type='xs:NMTOKEN' use='optional'/>
      <xs:attribute name='to' type='xs:string' use='optional'/>
      <xs:attribute name='version' type='xs:decimal' use='optional'/>
      <xs:attribute ref='xml:lang' use='optional'/>
    </xs:complexType>
  </xs:element>

  <xs:element name='features'>
    <xs:complexType>
      <xs:all xmlns:tls='urn:ietf:params:xml:ns:xmpp-tls'
              xmlns:sasl='urn:ietf:params:xml:ns:xmpp-sasl'
              xmlns:bind='urn:ietf:params:xml:ns:xmpp-bind'
              xmlns:sess='urn:ietf:params:xml:ns:xmpp-session'>
        <xs:element ref='tls:starttls' minOccurs='0'/>
        <xs:element ref='sasl:mechanisms' minOccurs='0'/>
        <xs:element ref='bind:bind' minOccurs='0'/>
        <xs:element ref='sess:session' minOccurs='0'/>
      </xs:all>
    </xs:complexType>
  </xs:element>

  <xs:element name='error'>
    <xs:complexType>
      <xs:sequence  xmlns:err='urn:ietf:params:xml:ns:xmpp-streams'>
        <xs:group   ref='err:streamErrorGroup'/>
        <xs:element ref='err:text'
                    minOccurs='0'
                    maxOccurs='1'/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

## C.2. Stream error namespace

TOC



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 493
Keysight, EX-1045
Page 493

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:ietf:params:xml:ns:xmpp-streams'
    xmlns='urn:ietf:params:xml:ns:xmpp-streams'
    elementFormDefault='qualified'>

  <xs:element name='bad-format' type='empty'/>
  <xs:element name='bad-namespace-prefix' type='empty'/>
  <xs:element name='conflict' type='empty'/>
  <xs:element name='connection-timeout' type='empty'/>
  <xs:element name='host-gone' type='empty'/>
  <xs:element name='host-unknown' type='empty'/>
  <xs:element name='improper-addressing' type='empty'/>
  <xs:element name='internal-server-error' type='empty'/>
  <xs:element name='invalid-from' type='empty'/>
  <xs:element name='invalid-id' type='empty'/>
  <xs:element name='invalid-namespace' type='empty'/>
  <xs:element name='invalid-xml' type='empty'/>
  <xs:element name='not-authorized' type='empty'/>
  <xs:element name='policy-violation' type='empty'/>
  <xs:element name='remote-connection-failed' type='empty'/>
  <xs:element name='resource-constraint' type='empty'/>
  <xs:element name='restricted-xml' type='empty'/>
  <xs:element name='see-other-host' type='xs:string'/>
  <xs:element name='system-shutdown' type='empty'/>
  <xs:element name='undefined-condition' type='empty'/>
  <xs:element name='unsupported-encoding' type='empty'/>
  <xs:element name='unsupported-stanza-type' type='empty'/>
  <xs:element name='unsupported-version' type='empty'/>
  <xs:element name='xml-not-well-formed' type='empty'/>

  <xs:group name='streamErrorGroup'>
    <xs:choice>
      <xs:element ref='bad-format'/>
      <xs:element ref='bad-namespace-prefix'/>
      <xs:element ref='conflict'/>
      <xs:element ref='connection-timeout'/>
      <xs:element ref='host-gone'/>
      <xs:element ref='host-unknown'/>
      <xs:element ref='improper-addressing'/>
      <xs:element ref='internal-server-error'/>
      <xs:element ref='invalid-from'/>
      <xs:element ref='invalid-id'/>
      <xs:element ref='invalid-namespace'/>
      <xs:element ref='invalid-xml'/>
      <xs:element ref='not-authorized'/>
      <xs:element ref='policy-violation'/>
      <xs:element ref='remote-connection-failed'/>
      <xs:element ref='resource-constraint'/>
      <xs:element ref='restricted-xml'/>
      <xs:element ref='see-other-host'/>
      <xs:element ref='system-shutdown'/>
      <xs:element ref='undefined-condition'/>
      <xs:element ref='unsupported-encoding'/>
      <xs:element ref='unsupported-stanza-type'/>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 494
Keysight, EX-1045
Page 494

```
      <xs:element ref='unsupported-version'/>
      <xs:element ref='xml-not-well-formed'/>
    </xs:choice>
  </xs:group>

  <xs:element name='text'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:string'>
          <xs:attribute ref='xml:lang' use='optional'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:simpleType name='empty'>
    <xs:restriction base='xs:string'>
      <xs:enumeration value=''/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

## C.3. TLS namespace

TOC

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:ietf:params:xml:ns:xmpp-tls'
    xmlns='urn:ietf:params:xml:ns:xmpp-tls'
    elementFormDefault='qualified'>

  <xs:element name='starttls'>
    <xs:complexType>
      <xs:sequence>
        <xs:element
            name='required'
            minOccurs='0'
            maxOccurs='1'
            type='empty'/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name='proceed' type='empty'/>
  <xs:element name='failure' type='empty'/>

  <xs:simpleType name='empty'>
    <xs:restriction base='xs:string'>
      <xs:enumeration value=''/>
    </xs:restriction>
  </xs:simpleType>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 495
Keysight, EX-1045
Page 495

```
</xs:schema>
```

## C.4. SASL namespace

TOC

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:ietf:params:xml:ns:xmpp-sasl'
    xmlns='urn:ietf:params:xml:ns:xmpp-sasl'
    elementFormDefault='qualified'>

  <xs:element name='mechanisms'>
    <xs:complexType>
      <xs:sequence>
        <xs:element name='mechanism'
                    maxOccurs='unbounded'
                    type='xs:string'/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name='auth'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='empty'>
          <xs:attribute name='mechanism'
                        type='xs:string'
                        use='optional'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name='challenge' type='xs:string'/>
  <xs:element name='response' type='xs:string'/>
  <xs:element name='abort' type='empty'/>
  <xs:element name='success' type='empty'/>

  <xs:element name='failure'>
    <xs:complexType>
      <xs:choice minOccurs='0'>
        <xs:element name='aborted' type='empty'/>
        <xs:element name='incorrect-encoding' type='empty'/>
        <xs:element name='invalid-authzid' type='empty'/>
        <xs:element name='invalid-mechanism' type='empty'/>
        <xs:element name='mechanism-too-weak' type='empty'/>
        <xs:element name='not-authorized' type='empty'/>
        <xs:element name='temporary-auth-failure' type='empty'/>
      </xs:choice>
    </xs:complexType>
  </xs:element>

  <xs:simpleType name='empty'>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 496
Keysight, EX-1045
Page 496

```
    <xs:restriction base='xs:string'>
      <xs:enumeration value=''/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

## C.5. Resource binding namespace

TOC

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:ietf:params:xml:ns:xmpp-bind'
    xmlns='urn:ietf:params:xml:ns:xmpp-bind'
    elementFormDefault='qualified'>

  <xs:element name='bind'>
    <xs:complexType>
      <xs:choice minOccurs='0' maxOccurs='1'>
        <xs:element name='resource' type='xs:string'/>
        <xs:element name='jid' type='xs:string'/>
      </xs:choice>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

## C.6. Dialback namespace

TOC

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='jabber:server:dialback'
    xmlns='jabber:server:dialback'
    elementFormDefault='qualified'>

  <xs:element name='result'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:token'>
          <xs:attribute name='from' type='xs:string' use='required'/>
          <xs:attribute name='to' type='xs:string' use='required'/>
          <xs:attribute name='type' use='optional'>
            <xs:simpleType>
              <xs:restriction base='xs:NCName'>
                <xs:enumeration value='invalid'/>
                <xs:enumeration value='valid'/>
              </xs:restriction>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 497
Keysight, EX-1045
Page 497

```
          </xs:simpleType>
        </xs:attribute>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

<xs:element name='verify'>
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base='xs:token'>
        <xs:attribute name='from' type='xs:string' use='required'/>
        <xs:attribute name='id' type='xs:NMTOKEN' use='required'/>
        <xs:attribute name='to' type='xs:string' use='required'/>
        <xs:attribute name='type' use='optional'>
          <xs:simpleType>
            <xs:restriction base='xs:NCName'>
              <xs:enumeration value='invalid'/>
              <xs:enumeration value='valid'/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

</xs:schema>
```

## C.7. Stanza error namespace

TOC

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:ietf:params:xml:ns:xmpp-stanzas'
    xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'
    elementFormDefault='qualified'>

  <xs:element name='bad-request' type='empty'/>
  <xs:element name='conflict' type='empty'/>
  <xs:element name='feature-not-implemented' type='empty'/>
  <xs:element name='forbidden' type='empty'/>
  <xs:element name='gone' type='xs:string'/>
  <xs:element name='internal-server-error' type='empty'/>
  <xs:element name='item-not-found' type='empty'/>
  <xs:element name='jid-malformed' type='empty'/>
  <xs:element name='not-acceptable' type='empty'/>
  <xs:element name='not-allowed' type='empty'/>
  <xs:element name='payment-required' type='empty'/>
  <xs:element name='recipient-unavailable' type='empty'/>
  <xs:element name='redirect' type='xs:string'/>
  <xs:element name='registration-required' type='empty'/>
  <xs:element name='remote-server-not-found' type='empty'/>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 498
Keysight, EX-1045
Page 498

```
  <xs:element name='remote-server-timeout' type='empty'/>
  <xs:element name='resource-constraint' type='empty'/>
  <xs:element name='service-unavailable' type='empty'/>
  <xs:element name='subscription-required' type='empty'/>
  <xs:element name='undefined-condition' type='empty'/>
  <xs:element name='unexpected-request' type='empty'/>

  <xs:group name='stanzaErrorGroup'>
    <xs:choice>
      <xs:element ref='bad-request'/>
      <xs:element ref='conflict'/>
      <xs:element ref='feature-not-implemented'/>
      <xs:element ref='forbidden'/>
      <xs:element ref='gone'/>
      <xs:element ref='internal-server-error'/>
      <xs:element ref='item-not-found'/>
      <xs:element ref='jid-malformed'/>
      <xs:element ref='not-acceptable'/>
      <xs:element ref='not-allowed'/>
      <xs:element ref='payment-required'/>
      <xs:element ref='recipient-unavailable'/>
      <xs:element ref='redirect'/>
      <xs:element ref='registration-required'/>
      <xs:element ref='remote-server-not-found'/>
      <xs:element ref='remote-server-timeout'/>
      <xs:element ref='resource-constraint'/>
      <xs:element ref='service-unavailable'/>
      <xs:element ref='subscription-required'/>
      <xs:element ref='undefined-condition'/>
      <xs:element ref='unexpected-request'/>
    </xs:choice>
  </xs:group>

  <xs:element name='text'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:string'>
          <xs:attribute ref='xml:lang' use='optional'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:simpleType name='empty'>
    <xs:restriction base='xs:string'>
      <xs:enumeration value=''/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

## Appendix D. Differences Between Core Jabber Protocols and XMPP

TOC

This section is non-normative.

XMPP has been adapted from the protocols originally developed in the Jabber open-source community, which can be thought of as "XMPP 0.9". Because there exists a large installed base of Jabber implementations and deployments, it may be helpful to specify the key differences between the relevant Jabber protocols and XMPP in order to expedite and encourage upgrades of those implementations and deployments to XMPP. This section summarizes the core differences, while the corresponding section of **[XMPP-IM]** summarizes the differences that relate specifically to instant messaging and presence applications.

## D.1. Channel Encryption

TOC

It was common practice in the Jabber community to use SSL for channel encryption on ports other than 5222 and 5269 (the convention is to use ports 5223 and 5270). XMPP uses TLS over the IANA-registered ports for channel encryption, as defined under **Use of TLS** herein.

## D.2. Authentication

TOC

The client-server authentication protocol developed in the Jabber community used a basic IQ interaction qualified by the 'jabber:iq:auth' namespace (documentation of this protocol is contained in **[JEP-0078]**, published by the **Jabber Software Foundation** [JSF]). XMPP uses SASL for authentication, as defined under **Use of SASL** herein.

The Jabber community did not develop an authentication protocol for server-to-server communications, only the **Server Dialback** protocol to prevent server spoofing. XMPP supersedes Server Dialback with a true server-to-server authentication protocol, as defined under **Use of SASL** herein.

## D.3. Resource Binding

TOC

Resource binding in the Jabber community was handled via the 'jabber:iq:auth' namespace (which was also used for client authentication with a server). XMPP defines a dedicated namespace for resource binding as well as the ability for a server to generate a resource identifier on behalf of a client, as defined under **Resource Binding**.

## D.4. JID Processing

TOC

JID processing was somewhat loosely defined by the Jabber community (documentation of forbidden characters and case handling is contained in **[JEP-0029]**, published by the **Jabber Software Foundation** [JSF]). XMPP specifies the use of **[NAMEPREP]** for domain identifiers and supplements Nameprep with two additional **[STRINGPREP]** profiles for JID processing: **Nodeprep** for node identifiers and **Resourceprep** for resource identifiers.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 500
Keysight, EX-1045
Page 500

### D.5. Error Handling

TOC

Stream-related errors were handled in the Jabber community via XML character data text in a <stream:error/> element. In XMPP, stream-related errors are handled via an extensible mechanism defined under **Stream Errors** herein.

Stanza-related errors were handled in the Jabber community via HTTP-style error codes. In XMPP, stanza-related errors are handled via an extensible mechanism defined under **Stanza Errors** herein. (Documentation of a mapping between Jabber and XMPP error handling mechanisms is contained in **[JEP-0086]**, published by the **Jabber Software Foundation** [JSF].)

---

### D.6. Internationalization

TOC

Although use of UTF-8 has always been standard practice within the Jabber community, the community did not define mechanisms for specifying the language of human-readable text provided in XML character data. XMPP specifies the use of the 'xml:lang' attribute in such contexts, as defined under **Stream Attributes** and **xml:lang** herein.

---

### D.7. Stream Version Attribute

TOC

The Jabber community did not include a 'version' attribute in stream headers. XMPP specifies inclusion of that attribute as a way to signal support for the stream features (authentication, encryption, etc.) defined under **Version Support** herein.

---

## Appendix E. Contributors

TOC

Most of the core aspects of the Extensible Messaging and Presence Protocol were developed originally within the Jabber open-source community in 1999. This community was founded by Jeremie Miller, who released source code for the initial version of the jabberd server in January 1999. Major early contributors to the base protocol also included Ryan Eatmon, Peter Millard, Thomas Muldowney, and Dave Smith. Work by the XMPP Working Group has concentrated especially on security and internationalization; in these areas, protocols for the use of TLS and SASL were originally contributed by Rob Norris, and stringprep profiles were originally contributed by Joe Hildebrand. The error code syntax was suggested by Lisa Dusseault.

---

## Appendix F. Acknowledgements

TOC

Thanks are due to a number of individuals in addition to the contributors listed. Although it is difficult to provide a complete list, the following individuals were particularly helpful in defining the protocols or in commenting on the specifications in this memo: Thomas Charron, Richard Dobson, Sam Hartman, Schuyler Heath, Jonathan Hogg, Cullen Jennings, Craig Kaes, Jacek Konieczny, Alexey Melnikov, Keith Minkler, Julian Missig, Pete Resnick, Marshall Rose, Alexey Shchepin, Jean-Louis Seguineau, Iain Shigeoka,



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 501
Keysight, EX-1045
Page 501

Greg Troxel, and David Waite. Thanks also to members of the XMPP Working Group and the IETF community for comments and feedback provided throughout the life of this memo.

## Author's Address

Peter Saint-Andre (editor)
Jabber Software Foundation
**Email:** stpeter@jabber.org

## Full Copyright Statement

Copyright © The Internet Society (2004).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at **http://www.ietf.org/ipr**.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at **ietf-ipr@ietf.org**.

## Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 502
**Keysight, EX-1045**
**Page 502**

| Network Working Group | P. Saint-Andre, Ed. |
|---|---|
| Request for Comments: 3921 | Jabber Software Foundation |
| Category: Standards Track | October 2004 |

RFC 3921 TOC


# Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence

## Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Copyright Notice

Copyright © The Internet Society (2004).

## Abstract

This memo describes extensions to and applications of the core features of the Extensible Messaging and Presence Protocol (XMPP) that provide the basic instant messaging (IM) and presence functionality defined in RFC 2779.

---

## Table of Contents

RFC 3921 TOC


1. Introduction
2. Syntax of XML Stanzas
3. Session Establishment
4. Exchanging Messages
5. Exchanging Presence Information
6. Managing Subscriptions
7. Roster Management
8. Integration of Roster Items and Presence Subscriptions
9. Subscription States
10. Blocking Communication
11. Server Rules for Handling XML Stanzas
12. IM and Presence Compliance Requirements
13. Internationalization Considerations
14. Security Considerations
15. IANA Considerations
16. References
    - 16.1. Normative References
    - 16.2. Informative References

Appendix A. vCards
Appendix B. XML Schemas
Appendix C. Differences Between Jabber IM/Presence Protocols and XMPP
Appendix D. Contributors
Appendix E. Acknowledgements
§ Author's Address




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 503
Keysight, EX-1045
Page 503

§ **Intellectual Property and Copyright Statements**

## 1. Introduction

TOC

### 1.1. Overview

TOC

The Extensible Messaging and Presence Protocol (XMPP) is a protocol for streaming **XML** [XML] elements in order to exchange messages and presence information in close to real time. The core features of XMPP are defined in **Extensible Messaging and Presence Protocol (XMPP): Core** [XMPP-CORE]. These features -- mainly XML streams, use of TLS and SASL, and the <message/>, <presence/>, and <iq/> children of the stream root -- provide the building blocks for many types of near-real-time applications, which may be layered on top of the core by sending application-specific data qualified by particular **XML namespaces** [XML-NAMES]. This memo describes extensions to and applications of the core features of XMPP that provide the basic functionality expected of an instant messaging (IM) and presence application as defined in **RFC 2779** [IMP-REQS].

### 1.2. Requirements

TOC

For the purposes of this memo, the requirements of a basic instant messaging and presence application are defined by **[IMP-REQS]**, which at a high level stipulates that a user must be able to complete the following use cases:

- Exchange messages with other users
- Exchange presence information with other users
- Manage subscriptions to and from other users
- Manage items in a contact list (in XMPP this is called a "roster")
- Block communications to or from specific other users

Detailed definitions of these functionality areas are contained in **[IMP-REQS]**, and the interested reader is directed to that document regarding the requirements addressed herein.

**[IMP-REQS]** also stipulates that presence services must be separable from instant messaging services; i.e., it must be possible to use the protocol to provide a presence service, an instant messaging service, or both. Although the text of this memo assumes that implementations and deployments will want to offer a unified instant messaging and presence service, there is no requirement that a service must offer both a presence service and an instant messaging service, and the protocol makes it possible to offer separate and distinct services for presence and for instant messaging.

Note: While XMPP-based instant messaging and presence meets the requirements of **[IMP-REQS]**, it was not designed explicitly with that specification in mind, since the base protocol evolved through an open development process within the Jabber open-source community before RFC 2779 was written. Note also that although protocols addressing many other functionality areas have been defined in the Jabber community,



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 504
Keysight, EX-1045
Page 504

such protocols are not included in this memo because they are not required by **[IMP-REQS]**.

### 1.3. Terminology

TOC

This memo inherits the terminology defined in **[XMPP-CORE]**.

The capitalized key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in **RFC 2119** [TERMS].

## 2. Syntax of XML Stanzas

TOC

The basic semantics and common attributes of XML stanzas qualified by the 'jabber:client' and 'jabber:server' namespaces are defined in **[XMPP-CORE]**. However, these namespaces also define various child elements, as well as values for the common 'type' attribute, that are specific to instant messaging and presence applications. Thus, before addressing particular "use cases" for such applications, we here further describe the syntax of XML stanzas, thereby supplementing the discussion in **[XMPP-CORE]**.

### 2.1. Message Syntax

TOC

Message stanzas qualified by the 'jabber:client' or 'jabber:server' namespace are used to "push" information to another entity. Common uses in instant messaging applications include single messages, messages sent in the context of a chat conversation, messages sent in the context of a multi-user chat room, headlines and other alerts, and errors.

#### 2.1.1. Types of Message

TOC

The 'type' attribute of a message stanza is RECOMMENDED; if included, it specifies the conversational context of the message, thus providing a hint regarding presentation (e.g., in a GUI). If included, the 'type' attribute MUST have one of the following values:

- chat -- The message is sent in the context of a one-to-one chat conversation. A compliant client SHOULD present the message in an interface enabling one-to-one chat between the two parties, including an appropriate conversation history.
- error -- An error has occurred related to a previous message sent by the sender (for details regarding stanza error syntax, refer to **[XMPP-CORE]**). A compliant client SHOULD present an appropriate interface informing the sender of the nature of the error.
- groupchat -- The message is sent in the context of a multi-user chat environment (similar to that of **[IRC]**). A compliant client SHOULD present the message in an interface enabling many-to-many chat between the parties, including a roster of parties in the chatroom and an appropriate



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 505
Keysight, EX-1045
Page 505

conversation history. Full definition of XMPP-based groupchat protocols is out of scope for this memo (for details see **[JEP-0045]**).

- headline -- The message is probably generated by an automated service that delivers or broadcasts content (news, sports, market information, RSS feeds, etc.). No reply to the message is expected, and a compliant client SHOULD present the message in an interface that appropriately differentiates the message from standalone messages, chat sessions, or groupchat sessions (e.g., by not providing the recipient with the ability to reply).
- normal -- The message is a single message that is sent outside the context of a one-to-one conversation or groupchat, and to which it is expected that the recipient will reply. A compliant client SHOULD present the message in an interface enabling the recipient to reply, but without a conversation history.

An IM application SHOULD support all of the foregoing message types; if an application receives a message with no 'type' attribute or the application does not understand the value of the 'type' attribute provided, it MUST consider the message to be of type "normal" (i.e., "normal" is the default). The "error" type MUST be generated only in response to an error related to a message received from another entity.

Although the 'type' attribute is OPTIONAL, it is considered polite to mirror the type in any replies to a message; furthermore, some specialized applications (e.g., a multi-user chat service) MAY at their discretion enforce the use of a particular message type (e.g., type='groupchat').

### 2.1.2. Child Elements

TOC

As described under **extended namespaces**, a message stanza MAY contain any properly-namespaced child element.

In accordance with the default namespace declaration, by default a message stanza is qualified by the 'jabber:client' or 'jabber:server' namespace, which defines certain allowable children of message stanzas. If the message stanza is of type "error", it MUST include an <error/> child; for details, see **[XMPP-CORE]**. Otherwise, the message stanza MAY contain any of the following child elements without an explicit namespace declaration:

1. <subject/>
2. <body/>
3. <thread/>

#### 2.1.2.1. Subject

TOC

The <subject/> element contains human-readable XML character data that specifies the topic of the message. The <subject/> element MUST NOT possess any attributes, with the exception of the 'xml:lang' attribute. Multiple instances of the <subject/> element MAY be included for the purpose of providing alternate versions of the same subject, but only if each instance possesses an 'xml:lang' attribute with a distinct language value. The <subject/> element MUST NOT contain mixed content (as defined in Section 3.2.2



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 506
Keysight, EX-1045
Page 506

of **[XML]**).

### 2.1.2.2. Body

TOC

The <body/> element contains human-readable XML character data that specifies the textual contents of the message; this child element is normally included but is OPTIONAL. The <body/> element MUST NOT possess any attributes, with the exception of the 'xml:lang' attribute. Multiple instances of the <body/> element MAY be included but only if each instance possesses an 'xml:lang' attribute with a distinct language value. The <body/> element MUST NOT contain mixed content (as defined in Section 3.2.2 of **[XML]**).

### 2.1.2.3. Thread

TOC

The <thread/> element contains non-human-readable XML character data specifying an identifier that is used for tracking a conversation thread (sometimes referred to as an "instant messaging session") between two entities. The value of the <thread/> element is generated by the sender and SHOULD be copied back in any replies. If used, it MUST be unique to that conversation thread within the stream and MUST be consistent throughout that conversation (a client that receives a message from the same full JID but with a different thread ID MUST assume that the message in question exists outside the context of the existing conversation thread). The use of the <thread/> element is OPTIONAL and is not used to identify individual messages, only conversations. A message stanza MUST NOT contain more than one <thread/> element. The <thread/> element MUST NOT possess any attributes. The value of the <thread/> element MUST be treated as opaque by entities; no semantic meaning may be derived from it, and only exact comparisons may be made against it. The <thread/> element MUST NOT contain mixed content (as defined in Section 3.2.2 of **[XML]**).

## 2.2. Presence Syntax

TOC

Presence stanzas are used qualified by the 'jabber:client' or 'jabber:server' namespace to express an entity's current network availability (offline or online, along with various sub-states of the latter and optional user-defined descriptive text), and to notify other entities of that availability. Presence stanzas are also used to negotiate and manage subscriptions to the presence of other entities.

### 2.2.1. Types of Presence

TOC

The 'type' attribute of a presence stanza is OPTIONAL. A presence stanza that does not possess a 'type' attribute is used to signal to the server that the sender is online and available for communication. If included, the 'type' attribute specifies a lack of availability, a request to manage a subscription to another entity's presence, a request for another entity's current presence, or an error related to a previously-sent presence stanza. If included, the 'type' attribute MUST have one of the following values:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 507
Keysight, EX-1045
Page 507

- unavailable -- Signals that the entity is no longer available for communication.
- subscribe -- The sender wishes to subscribe to the recipient's presence.
- subscribed -- The sender has allowed the recipient to receive their presence.
- unsubscribe -- The sender is unsubscribing from another entity's presence.
- unsubscribed -- The subscription request has been denied or a previously-granted subscription has been cancelled.
- probe -- A request for an entity's current presence; SHOULD be generated only by a server on behalf of a user.
- error -- An error has occurred regarding processing or delivery of a previously-sent presence stanza.

For detailed information regarding presence semantics and the subscription model used in the context of XMPP-based instant messaging and presence applications, refer to **Exchanging Presence Information** and **Managing Subscriptions**.

---

### 2.2.2. Child Elements

TOC

As described under **extended namespaces**, a presence stanza MAY contain any properly-namespaced child element.

In accordance with the default namespace declaration, by default a presence stanza is qualified by the 'jabber:client' or 'jabber:server' namespace, which defines certain allowable children of presence stanzas. If the presence stanza is of type "error", it MUST include an <error/> child; for details, see **[XMPP-CORE]**. If the presence stanza possesses no 'type' attribute, it MAY contain any of the following child elements (note that the <status/> child MAY be sent in a presence stanza of type "unavailable" or, for historical reasons, "subscribe"):

1. <show/>
2. <status/>
3. <priority/>

---

#### 2.2.2.1. Show

TOC

The OPTIONAL <show/> element contains non-human-readable XML character data that specifies the particular availability status of an entity or specific resource. A presence stanza MUST NOT contain more than one <show/> element. The <show/> element MUST NOT possess any attributes. If provided, the XML character data value MUST be one of the following (additional availability types could be defined through a properly-namespaced child element of the presence stanza):

- away -- The entity or resource is temporarily away.
- chat -- The entity or resource is actively interested in chatting.
- dnd -- The entity or resource is busy (dnd = "Do Not Disturb").
- xa -- The entity or resource is away for an extended period (xa = "eXtended Away").

If no <show/> element is provided, the entity is assumed to be online and available.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 508
Keysight, EX-1045
Page 508

#### 2.2.2.2. Status

TOC

The OPTIONAL <status/> element contains XML character data specifying a natural-language description of availability status. It is normally used in conjunction with the show element to provide a detailed description of an availability state (e.g., "In a meeting"). The <status/> element MUST NOT possess any attributes, with the exception of the 'xml:lang' attribute. Multiple instances of the <status/> element MAY be included but only if each instance possesses an 'xml:lang' attribute with a distinct language value.

#### 2.2.2.3. Priority

TOC

The OPTIONAL <priority/> element contains non-human-readable XML character data that specifies the priority level of the resource. The value MUST be an integer between -128 and +127. A presence stanza MUST NOT contain more than one <priority/> element. The <priority/> element MUST NOT possess any attributes. If no priority is provided, a server SHOULD consider the priority to be zero. For information regarding the semantics of priority values in stanza routing within instant messaging and presence applications, refer to **Server Rules for Handling XML Stanzas**.

### 2.3. IQ Syntax

TOC

IQ stanzas provide a structured request-response mechanism. The basic semantics of that mechanism (e.g., that the 'id' attribute is REQUIRED) are defined in **[XMPP-CORE]**, whereas the specific semantics required to complete particular use cases are defined in all cases by an **extended namespace** (note that the 'jabber:client' and 'jabber:server' namespaces do not define any children of IQ stanzas other than the common <error/>). This memo defines two such extended namespaces, one for **Roster Management** and the other for **Blocking Communication**; however, an IQ stanza MAY contain structured information qualified by any extended namespace.

### 2.4. Extended Namespaces

TOC

While the three XML stanza kinds defined in the "jabber:client" or "jabber:server" namespace (along with their attributes and child elements) provide a basic level of functionality for messaging and presence, XMPP uses XML namespaces to extend the stanzas for the purpose of providing additional functionality. Thus a message or presence stanza MAY contain one or more optional child elements specifying content that extends the meaning of the message (e.g., an XHTML-formatted version of the message body), and an IQ stanza MAY contain one such child element. This child element MAY have any name and MUST possess an 'xmlns' namespace declaration (other than "jabber:client", "jabber:server", or "http://etherx.jabber.org/streams") that defines all data contained within the child element.

Support for any given extended namespace is OPTIONAL on the part of any implementation (aside from the extended namespaces defined herein). If an entity does



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 509
Keysight, EX-1045
Page 509

not understand such a namespace, the entity's expected behavior depends on whether the entity is (1) the recipient or (2) an entity that is routing the stanza to the recipient:

Recipient:

If a recipient receives a stanza that contains a child element it does not understand, it SHOULD ignore that specific XML data, i.e., it SHOULD not process it or present it to a user or associated application (if any). In particular:

- If an entity receives a message or presence stanza that contains XML data qualified by a namespace it does not understand, the portion of the stanza that is in the unknown namespace SHOULD be ignored.
- If an entity receives a message stanza whose only child element is qualified by a namespace it does not understand, it MUST ignore the entire stanza.
- If an entity receives an IQ stanza of type "get" or "set" containing a child element qualified by a namespace it does not understand, the entity SHOULD return an IQ stanza of type "error" with an error condition of <service-unavailable/>.

Router:

If a routing entity (usually a server) handles a stanza that contains a child element it does not understand, it SHOULD ignore the associated XML data by passing it on untouched to the recipient.

---

## 3. Session Establishment

TOC

Most instant messaging and presence applications based on XMPP are implemented via a client-server architecture that requires a client to establish a session on a server in order to engage in the expected instant messaging and presence activities. However, there are several pre-conditions that MUST be met before a client can establish an instant messaging and presence session. These are:

1. Stream Authentication -- a client MUST complete stream authentication as documented in **[XMPP-CORE]** before attempting to establish a session or send any XML stanzas.
2. Resource Binding -- after completing stream authentication, a client MUST bind a resource to the stream so that the client's address is of the form <user@domain/resource>, after which the entity is now said to be a "connected resource" in the terminology of **[XMPP-CORE]**.

If a server supports sessions, it MUST include a <session/> element qualified by the 'urn:ietf:params:xml:ns:xmpp-session' namespace in the stream features it advertises to a client after the completion of stream authentication as defined in **[XMPP-CORE]**:

Server advertises session establishment feature to client:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    id='c2s_345'
```

```
    from='example.com'
    version='1.0'>
<stream:features>
  <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'/>
  <session xmlns='urn:ietf:params:xml:ns:xmpp-session'/>
</stream:features>
```

Upon being so informed that session establishment is required (and after completing resource binding), the client MUST establish a session if it desires to engage in instant messaging and presence functionality; it completes this step by sending to the server an IQ stanza of type "set" containing an empty <session/> child element qualified by the 'urn:ietf:params:xml:ns:xmpp-session' namespace:

Step 1: Client requests session with server:

```
<iq to='example.com'
    type='set'
    id='sess_1'>
  <session xmlns='urn:ietf:params:xml:ns:xmpp-session'/>
</iq>
```

Step 2: Server informs client that session has been created:

```
<iq from='example.com'
    type='result'
    id='sess_1'/>
```

Upon establishing a session, a connected resource (in the terminology of **[XMPP-CORE]**) is said to be an "active resource".

Several error conditions are possible. For example, the server may encounter an internal condition that prevents it from creating the session, the username or authorization identity may lack permissions to create a session, or there may already be an active resource associated with a resource identifier of the same name.

If the server encounters an internal condition that prevents it from creating the session, it MUST return an error.

Step 2 (alt): Server responds with error (internal server error):

```
<iq from='example.com' type='error' id='sess_1'>
  <session xmlns='urn:ietf:params:xml:ns:xmpp-session'/>
  <error type='wait'>
    <internal-server-error
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

If the username or resource is not allowed to create a session, the server MUST return an error (e.g., forbidden).



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 511
Keysight, EX-1045
Page 511

Step 2 (alt): Server responds with error (username or resource not allowed to create session):

```
<iq from='example.com' type='error' id='sess_1'>
  <session xmlns='urn:ietf:params:xml:ns:xmpp-session'/>
  <error type='auth'>
    <forbidden
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

If there is already an active resource of the same name, the server MUST either (1) terminate the active resource and allow the newly-requested session, or (2) disallow the newly-requested session and maintain the active resource. Which of these the server does is up to the implementation, although it is RECOMMENDED to implement case #1. In case #1, the server SHOULD send a <conflict/> stream error to the active resource, terminate the XML stream and underlying TCP connection for the active resource, and return a IQ stanza of type "result" (indicating success) to the newly-requested session. In case #2, the server SHOULD send a <conflict/> stanza error to the newly-requested session but maintain the XML stream for that connection so that the newly-requested session has an opportunity to negotiate a non-conflicting resource identifier before sending another request for session establishment.

Step 2 (alt): Server informs existing active resource of resource conflict (case #1):

```
<stream:error>
  <conflict xmlns='urn:ietf:params:xml:ns:xmpp-streams'/>
</stream:error>
</stream:stream>
```

Step 2 (alt): Server informs newly-requested session of resource conflict (case #2):

```
<iq from='example.com' type='error' id='sess_1'>
  <session xmlns='urn:ietf:params:xml:ns:xmpp-session'/>
  <error type='cancel'>
    <conflict xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

After establishing a session, a client SHOULD send initial presence and request its roster as described below, although these actions are OPTIONAL.

Note: Before allowing the creation of instant messaging and presence sessions, a server MAY require prior account provisioning. Possible methods for account provisioning include account creation by a server administrator as well as in-band account registration using the 'jabber:iq:register' namespace; the latter method is out of scope for this memo, but is documented in **[JEP-0077]**, published by the **Jabber Software Foundation** [JSF].

## 4. Exchanging Messages

TOC

Exchanging messages is a basic use of XMPP and is brought about when a user generates a message stanza that is addressed to another entity. As defined under **Server Rules for Handling XML Stanzas**, the sender's server is responsible for delivering the message to the intended recipient (if the recipient is on the same server) or for routing the message to the recipient's server (if the recipient is on a different server).

For information regarding the syntax of message stanzas as well as their defined attributes and child elements, refer to **Message Syntax**.

### 4.1. Specifying an Intended Recipient

TOC

An instant messaging client SHOULD specify an intended recipient for a message by providing the JID of an entity other than the sender in the 'to' attribute of the <message/> stanza. If the message is being sent in reply to a message previously received from an address of the form <user@domain/resource> (e.g., within the context of a chat session), the value of the 'to' address SHOULD be of the form <user@domain/resource> rather than of the form <user@domain> unless the sender has knowledge (via presence) that the intended recipient's resource is no longer available. If the message is being sent outside the context of any existing chat session or received message, the value of the 'to' address SHOULD be of the form <user@domain> rather than of the form <user@domain/resource>.

### 4.2. Specifying a Message Type

TOC

As noted, it is RECOMMENDED for a message stanza to possess a 'type' attribute whose value captures the conversational context (if any) of the message (see **Type**).

The following example shows a valid value of the 'type' attribute:

Example: A message of a defined type:

```
<message
    to='romeo@example.net'
    from='juliet@example.com/balcony'
    type='chat'
    xml:lang='en'>
  <body>Wherefore art thou, Romeo?</body>
</message>
```

### 4.3. Specifying a Message Body

TOC

A message stanza MAY (and often will) contain a child <body/> element whose XML character data specifies the primary meaning of the message (see **Body**).

Example: A message with a body:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 513
Keysight, EX-1045
Page 513

```
<message
    to='romeo@example.net'
    from='juliet@example.com/balcony'
    type='chat'
    xml:lang='en'>
  <body>Wherefore art thou, Romeo?</body>
  <body xml:lang='cz'>Pro&#x010D;e&#x017D; jsi ty, Romeo?</body>
</message>
```

## 4.4. Specifying a Message Subject

TOC

A message stanza MAY contain one or more child <subject/> elements specifying the topic of the message (see **Subject**).

Example: A message with a subject:

```
<message
    to='romeo@example.net'
    from='juliet@example.com/balcony'
    type='chat'
    xml:lang='en'>
  <subject>I implore you!</subject>
  <subject
      xml:lang='cz'>&#x00DA;p&#x011B;nliv&#x011B; prosim!</subject>
  <body>Wherefore art thou, Romeo?</body>
  <body xml:lang='cz'>Pro&#x010D;e&#x017D; jsi ty, Romeo?</body>
</message>
```

## 4.5. Specifying a Conversation Thread

TOC

A message stanza MAY contain a child <thread/> element specifying the conversation thread in which the message is situated, for the purpose of tracking the conversation (see **Thread**).

Example: A threaded conversation:

```
<message
    to='romeo@example.net/orchard'
    from='juliet@example.com/balcony'
    type='chat'
    xml:lang='en'>
  <body>Art thou not Romeo, and a Montague?</body>
  <thread>e0ffe42b28561960c6b12b944a092794b9683a38</thread>
</message>

<message
    to='juliet@example.com/balcony'
    from='romeo@example.net/orchard'
    type='chat'
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 514
Keysight, EX-1045
Page 514

```
    xml:lang='en'>
  <body>Neither, fair saint, if either thee dislike.</body>
  <thread>e0ffe42b28561960c6b12b944a092794b9683a38</thread>
</message>

<message
    to='romeo@example.net/orchard'
    from='juliet@example.com/balcony'
    type='chat'
    xml:lang='en'>
  <body>How cam'st thou hither, tell me, and wherefore?</body>
  <thread>e0ffe42b28561960c6b12b944a092794b9683a38</thread>
</message>
```

## 5. Exchanging Presence Information

TOC

Exchanging presence information is made relatively straightforward within XMPP by using presence stanzas. However, we see here a contrast to the handling of messages: although a client MAY send directed presence information to another entity by including a 'to' address, normally presence notifications (i.e., presence stanzas with no 'type' or of type "unavailable" and with no 'to' address) are sent from a client to its server and then broadcasted by the server to any entities that are subscribed to the presence of the sending entity (in the terminology of **RFC 2778** [IMP-MODEL], these entities are subscribers). This broadcast model does not apply to subscription-related presence stanzas or presence stanzas of type "error", but to presence notifications only as defined above. (Note: While presence information MAY be provided on a user's behalf by an automated service, normally it is provided by the user's client.)

For information regarding the syntax of presence stanzas as well as their defined attributes and child elements, refer to **[XMPP-CORE]**.

### 5.1. Client and Server Presence Responsibilities

TOC

#### 5.1.1. Initial Presence

TOC

After establishing a session, a client SHOULD send initial presence to the server in order to signal its availability for communications. As defined herein, the initial presence stanza (1) MUST possess no 'to' address (signalling that it is meant to be broadcasted by the server on behalf of the client) and (2) MUST possess no 'type' attribute (signalling the user's availability). After sending initial presence, an active resource is said to be an "available resource".

Upon receiving initial presence from a client, the user's server MUST do the following if there is not already one or more available resources for the user (if there is already one or more available resources for the user, the server obviously does not need to send the presence probes, since it already possesses the requisite information):

1. Send presence probes (i.e., presence stanzas whose 'type' attribute is set to

a value of "probe") from the full JID (e.g., <user@example.com/resource>) of the user to all contacts to which the user is subscribed in order to determine if they are available; such contacts are those for which a JID is present in the user's roster with the 'subscription' attribute set to a value of "to" or "both". (Note: The user's server MUST NOT send presence probes to contacts from which the user is blocking inbound presence notifications, as described under **Blocking Inbound Presence Notifications**.)

2. Broadcast initial presence from the full JID (e.g., <user@example.com/resource>) of the user to all contacts that are subscribed to the user's presence information; such contacts are those for which a JID is present in the user's roster with the 'subscription' attribute set to a value of "from" or "both". (Note: The user's server MUST NOT broadcast initial presence to contacts to which the user is blocking outbound presence notifications, as described under **Blocking Outbound Presence Notifications**.)

In addition, the user's server MUST broadcast initial presence from the user's new available resource to any of the user's existing available resources (if any).

Upon receiving initial presence from the user, the contact's server MUST deliver the user's presence stanza to the full JIDs (<contact@example.org/resource>) associated with all of the contact's available resources, but only if the user is in the contact's roster with a subscription state of "to" or "both" and the contact has not blocked inbound presence notifications from the user's bare or full JID (as defined under **Blocking Inbound Presence Notifications**).

If the user's server receives a presence stanza of type "error" in response to the initial presence that it sent to a contact on behalf of the user, it SHOULD NOT send further presence updates to that contact (until and unless it receives a presence stanza from the contact).

---

### 5.1.2. Presence Broadcast

TOC

After sending initial presence, the user MAY update its presence information for broadcasting at any time during its session by sending a presence stanza with no 'to' address and either no 'type' attribute or a 'type' attribute with a value of "unavailable". (Note: A user's client SHOULD NOT send a presence update to broadcast information that changes independently of the user's presence and availability.)

If the presence stanza lacks a 'type' attribute (i.e., expresses availability), the user's server MUST broadcast the full XML of that presence stanza to all contacts (1) that are in the user's roster with a subscription type of "from" or "both", (2) to whom the user has not blocked outbound presence notifications, and (3) from whom the server has not received a presence error during the user's session (as well as to any of the user's other available resources).

If the presence stanza has a 'type' attribute set to a value of "unavailable", the user's server MUST broadcast the full XML of that presence stanza to all entities that fit the above description, as well as to any entities to which the user has sent directed available presence during the user's session (if the user has not yet sent directed unavailable presence to that entity).



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 516
Keysight, EX-1045
Page 516

### 5.1.3. Presence Probes

TOC

Upon receiving a presence probe from the user, the contact's server SHOULD reply as follows:

1. If the user is not in the contact's roster with a subscription state of "From", "From + Pending Out", or "Both" (as defined under **Subscription States**), the contact's server MUST return a presence stanza of type "error" in response to the presence probe (however, if a server receives a presence probe from a subdomain of the server's hostname or another such trusted service, it MAY provide presence information about the user to that entity). Specifically:
    - if the user is in the contact's roster with a subscription state of "None", "None + Pending Out", or "To" (or is not in the contact's roster at all), the contact's server MUST return a <forbidden/> stanza error in response to the presence probe.
    - if the user is in the contact's roster with a subscription state of "None + Pending In", "None + Pending Out/In", or "To + Pending In", the contact's server MUST return a <not-authorized/> stanza error in response to the presence probe.
2. Else, if the contact is blocking presence notifications to the user's bare JID or full JID (using either a default list or active list as defined under **Blocking Outbound Presence Notifications**), the server MUST NOT reply to the presence probe.
3. Else, if the contact has no available resources, the server MUST either (1) reply to the presence probe by sending to the user the full XML of the last presence stanza of type "unavailable" received by the server from the contact, or (2) not reply at all.
4. Else, if the contact has at least one available resource, the server MUST reply to the presence probe by sending to the user the full XML of the last presence stanza with no 'to' attribute received by the server from each of the contact's available resources (again, subject to privacy lists in force for each session).

### 5.1.4. Directed Presence

TOC

A user MAY send directed presence to another entity (i.e., a presence stanza with a 'to' attribute whose value is the JID of the other entity and with either no 'type' attribute or a 'type' attribute whose value is "unavailable"). There are three possible cases:

1. If the user sends directed presence to a contact that is in the user's roster with a subscription type of "from" or "both" after having sent initial presence and before sending unavailable presence broadcast, the user's server MUST route or deliver the full XML of that presence stanza (subject to privacy lists) but SHOULD NOT otherwise modify the contact's status regarding presence broadcast (i.e., it SHOULD include the contact's JID in any subsequent presence broadcasts initiated by the user).
2. If the user sends directed presence to an entity that is not in the user's roster with a subscription type of "from" or "both" after having sent initial



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 517
Keysight, EX-1045
Page 517

presence and before sending unavailable presence broadcast, the user's server MUST route or deliver the full XML of that presence stanza to the entity but MUST NOT modify the contact's status regarding available presence broadcast (i.e., it MUST NOT include the entity's JID in any subsequent broadcasts of available presence initiated by the user); however, if the available resource from which the user sent the directed presence become unavailable, the user's server MUST broadcast that unavailable presence to the entity (if the user has not yet sent directed unavailable presence to that entity).

3. If the user sends directed presence without first sending initial presence or after having sent unavailable presence broadcast (i.e., the resource is active but not available), the user's server MUST treat the entities to which the user sends directed presence in the same way that it treats the entities listed in case #2 above.

### 5.1.5. Unavailable Presence

TOC

Before ending its session with a server, a client SHOULD gracefully become unavailable by sending a final presence stanza that possesses no 'to' attribute and that possesses a 'type' attribute whose value is "unavailable" (optionally, the final presence stanza MAY contain one or more <status/> elements specifying the reason why the user is no longer available). However, the user's server MUST NOT depend on receiving final presence from an available resource, since the resource may become unavailable unexpectedly or may be timed out by the server. If one of the user's resources becomes unavailable for any reason (either gracefully or ungracefully), the user's server MUST broadcast unavailable presence to all contacts (1) that are in the user's roster with a subscription type of "from" or "both", (2) to whom the user has not blocked outbound presence, and (3) from whom the server has not received a presence error during the user's session; the user's server MUST also send that unavailable presence stanza to any of the user's other available resources, as well as to any entities to which the user has sent directed presence during the user's session for that resource (if the user has not yet sent directed unavailable presence to that entity). Any presence stanza with no 'type' attribute and no 'to' attribute that is sent after sending directed unavailable presence or broadcasted unavailable presence MUST be broadcasted by the server to all subscribers.

### 5.1.6. Presence Subscriptions

TOC

A subscription request is a presence stanza whose 'type' attribute has a value of "subscribe". If the subscription request is being sent to an instant messaging contact, the JID supplied in the 'to' attribute SHOULD be of the form <contact@example.org> rather than <contact@example.org/resource>, since the desired result is normally for the user to receive presence from all of the contact's resources, not merely the particular resource specified in the 'to' attribute.

A user's server MUST NOT automatically approve subscription requests on the user's behalf. All subscription requests MUST be directed to the user's client, specifically to one or more available resources associated with the user. If there is no available resource associated with the user when the subscription request is received by the user's server, the user's server MUST keep a record of the subscription request and deliver the request



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 518
Keysight, EX-1045
Page 518

when the user next creates an available resource, until the user either approves or denies the request. If there is more than one available resource associated with the user when the subscription request is received by the user's server, the user's server MUST broadcast that subscription request to all available resources in accordance with **Server Rules for Handling XML Stanzas**. (Note: If an active resource has not provided initial presence, the server MUST NOT consider it to be available and therefore MUST NOT send subscription requests to it.) However, if the user receives a presence stanza of type "subscribe" from a contact to whom the user has already granted permission to see the user's presence information (e.g., in cases when the contact is seeking to resynchronize subscription states), the user's server SHOULD auto-reply on behalf of the user. In addition, the user's server MAY choose to re-send an unapproved pending subscription request to the contact based on an implementation-specific algorithm (e.g., whenever a new resource becomes available for the user, or after a certain amount of time has elapsed); this helps to recover from transient, silent errors that may have occurred in relation to the original subscription request.

## 5.2. Specifying Availability Status

TOC

A client MAY provide further information about its availability status by using the <show/> element (see **Show**).

Example: Availability status:

```
<presence>
  <show>dnd</show>
</presence>
```

## 5.3. Specifying Detailed Status Information

TOC

In conjunction with the <show/> element, a client MAY provide detailed status information by using the <status/> element (see **Status**).

Example: Detailed status information:

```
<presence xml:lang='en'>
  <show>dnd</show>
  <status>Wooing Juliet</status>
  <status xml:lang='cz'>Ja dvo&#x0159;&#x00ED;m Juliet</status>
</presence>
```

## 5.4. Specifying Presence Priority

TOC

A client MAY provide a priority for its resource by using the <priority/> element (see **Priority**).

Example: Presence priority:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 519
Keysight, EX-1045
Page 519

```
<presence xml:lang='en'>
  <show>dnd</show>
  <status>Wooing Juliet</status>
  <status xml:lang='cz'>Ja dvo&#x0159;&#x00ED;m Juliet</status>
  <priority>1</priority>
</presence>
```

## 5.5. Presence Examples

TOC

The examples in this section illustrate the presence-related protocols described above. The user is romeo@example.net, he has an available resource whose resource identifier is "orchard", and he has the following individuals in his roster:

- juliet@example.com (subscription="both" and she has two available resources, one whose resource is "chamber" and another whose resource is "balcony")
- benvolio@example.org (subscription="to")
- mercutio@example.org (subscription="from")

Example 1: User sends initial presence:

```
<presence/>
```

Example 2: User's server sends presence probes to contacts with subscription="to" and subscription="both" on behalf of the user's available resource:

```
<presence
    type='probe'
    from='romeo@example.net/orchard'
    to='juliet@example.com'/>

<presence
    type='probe'
    from='romeo@example.net/orchard'
    to='benvolio@example.org'/>
```

Example 3: User's server sends initial presence to contacts with subscription="from" and subscription="both" on behalf of the user's available resource:

```
<presence
    from='romeo@example.net/orchard'
    to='juliet@example.com'/>

<presence
    from='romeo@example.net/orchard'
    to='mercutio@example.org'/>
```

Example 4: Contacts' servers reply to presence probe on behalf of all available



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 520
Keysight, EX-1045
Page 520

resources:

```
<presence
    from='juliet@example.com/balcony'
    to='romeo@example.net/orchard'
    xml:lang='en'>
  <show>away</show>
  <status>be right back</status>
  <priority>0</priority>
</presence>

<presence
    from='juliet@example.com/chamber'
    to='romeo@example.net/orchard'>
  <priority>1</priority>
</presence>

<presence
    from='benvolio@example.org/pda'
    to='romeo@example.net/orchard'
    xml:lang='en'>
  <show>dnd</show>
  <status>gallivanting</status>
</presence>
```

Example 5: Contacts' servers deliver user's initial presence to all available resources or return error to user:

```
<presence
    from='romeo@example.net/orchard'
    to='juliet@example.com/chamber'/>

<presence
    from='romeo@example.net/orchard'
    to='juliet@example.com/balcony'/>

<presence
    type='error'
    from='mercutio@example.org'
    to='romeo@example.net/orchard'>
  <error type='cancel'>
    <gone xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</presence>
```

Example 6: User sends directed presence to another user not in his roster:

```
<presence
    from='romeo@example.net/orchard'
    to='nurse@example.com'
    xml:lang='en'>
  <show>dnd</show>
  <status>courting Juliet</status>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 521
Keysight, EX-1045
Page 521

```
  <priority>0</priority>
</presence>
```

Example 7: User sends updated available presence information for broadcasting:

```
<presence xml:lang='en'>
  <show>away</show>
  <status>I shall return!</status>
  <priority>1</priority>
</presence>
```

Example 8: User's server broadcasts updated presence information only to one contact (not those from whom an error was received or to whom the user sent directed presence):

```
<presence
    from='romeo@example.net/orchard'
    to='juliet@example.com'
    xml:lang='en'>
  <show>away</show>
  <status>I shall return!</status>
  <priority>1</priority>
</presence>
```

Example 9: Contact's server delivers updated presence information to all of the contact's available resources:

```
[to "balcony" resource...]
<presence
    from='romeo@example.net/orchard'
    to='juliet@example.com'
    xml:lang='en'>
  <show>away</show>
  <status>I shall return!</status>
  <priority>1</priority>
</presence>

[to "chamber" resource...]
<presence
    from='romeo@example.net/orchard'
    to='juliet@example.com'
    xml:lang='en'>
  <show>away</show>
  <status>I shall return!</status>
  <priority>1</priority>
</presence>
```

Example 10: One of the contact's resources broadcasts final presence:

```
<presence from='juliet@example.com/balcony' type='unavailable'/>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 522
Keysight, EX-1045
Page 522

Example 11: Contact's server sends unavailable presence information to user:

```
<presence
    type='unavailable'
    from='juliet@example.com/balcony'
    to='romeo@example.net/orchard'/>
```

Example 12: User sends final presence:

```
<presence from='romeo@example.net/orchard'
          type='unavailable'
          xml:lang='en'>
  <status>gone home</status>
</presence>
```

Example 13: User's server broadcasts unavailable presence information to contact as well as to the person to whom the user sent directed presence:

```
<presence
    type='unavailable'
    from='romeo@example.net/orchard'
    to='juliet@example.com'
    xml:lang='en'>
  <status>gone home</status>
</presence>

<presence
    from='romeo@example.net/orchard'
    to='nurse@example.com'
    xml:lang='en'>
  <status>gone home</status>
</presence>
```

## 6. Managing Subscriptions

TOC

In order to protect the privacy of instant messaging users and any other entities, presence and availability information is disclosed only to other entities that the user has approved. When a user has agreed that another entity may view its presence, the entity is said to have a subscription to the user's presence information. A subscription lasts across sessions; indeed, it lasts until the subscriber unsubscribes or the subscribee cancels the previously-granted subscription. Subscriptions are managed within XMPP by sending presence stanzas containing specially-defined attributes.

Note: There are important interactions between subscriptions and rosters; these are defined under **Integration of Roster Items and Presence Subscriptions**, and the reader must refer to that section for a complete understanding of presence subscriptions.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 523
Keysight, EX-1045
Page 523

### 6.1. Requesting a Subscription

TOC

A request to subscribe to another entity's presence is made by sending a presence stanza of type "subscribe".

Example: Sending a subscription request:

```
<presence to='juliet@example.com' type='subscribe'/>
```

For client and server responsibilities regarding presence subscription requests, refer to **Presence Subscriptions**.

---

### 6.2. Handling a Subscription Request

TOC

When a client receives a subscription request from another entity, it MUST either approve the request by sending a presence stanza of type "subscribed" or refuse the request by sending a presence stanza of type "unsubscribed".

Example: Approving a subscription request:

```
<presence to='romeo@example.net' type='subscribed'/>
```

Example: Refusing a presence subscription request:

```
<presence to='romeo@example.net' type='unsubscribed'/>
```

---

### 6.3. Cancelling a Subscription from Another Entity

TOC

If a user would like to cancel a previously-granted subscription request, it sends a presence stanza of type "unsubscribed".

Example: Cancelling a previously granted subscription request:

```
<presence to='romeo@example.net' type='unsubscribed'/>
```

---

### 6.4. Unsubscribing from Another Entity's Presence

TOC

If a user would like to unsubscribe from the presence of another entity, it sends a presence stanza of type "unsubscribe".

Example: Unsubscribing from an entity's presence:

```
<presence to='juliet@example.com' type='unsubscribe'/>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 524
Keysight, EX-1045
Page 524

## 7. Roster Management

TOC

In XMPP, one's contact list is called a roster, which consists of any number of specific roster items, each roster item being identified by a unique JID (usually of the form <contact@domain>). A user's roster is stored by the user's server on the user's behalf so that the user may access roster information from any resource.

Note: There are important interactions between rosters and subscriptions; these are defined under **Integration of Roster Items and Presence Subscriptions**, and the reader must refer to that section for a complete understanding of roster management.

### 7.1. Syntax and Semantics

TOC

Rosters are managed using IQ stanzas, specifically by means of a <query/> child element qualified by the 'jabber:iq:roster' namespace. The <query/> element MAY contain one or more <item/> children, each describing a unique roster item or "contact".

The "key" or unique identifier for each roster item is a JID, encapsulated in the 'jid' attribute of the <item/> element (which is REQUIRED). The value of the 'jid' attribute SHOULD be of the form <user@domain> if the item is associated with another (human) instant messaging user.

The state of the presence subscription in relation to a roster item is captured in the 'subscription' attribute of the <item/> element. Allowable values for this attribute are:

- "none" -- the user does not have a subscription to the contact's presence information, and the contact does not have a subscription to the user's presence information
- "to" -- the user has a subscription to the contact's presence information, but the contact does not have a subscription to the user's presence information
- "from" -- the contact has a subscription to the user's presence information, but the user does not have a subscription to the contact's presence information
- "both" -- both the user and the contact have subscriptions to each other's presence information

Each <item/> element MAY contain a 'name' attribute, which sets the "nickname" to be associated with the JID, as determined by the user (not the contact). The value of the 'name' attribute is opaque.

Each <item/> element MAY contain one or more <group/> child elements, for use in collecting roster items into various categories. The XML character data of the <group/> element is opaque.

### 7.2. Business Rules

TOC

A server MUST ignore any 'to' address on a roster "set", and MUST treat any roster "set"



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 525
Keysight, EX-1045
Page 525

as applying to the sender. For added safety, a client SHOULD check the "from" address of a "roster push" (incoming IQ of type "set" containing a roster item) to ensure that it is from a trusted source; specifically, the stanza MUST either have no 'from' attribute (i.e., implicitly from the server) or have a 'from' attribute whose value matches the user's bare JID (of the form <user@domain>) or full JID (of the form <user@domain/resource>); otherwise, the client SHOULD ignore the "roster push".

## 7.3. Retrieving One's Roster on Login

TOC

Upon connecting to the server and becoming an active resource, a client SHOULD request the roster before sending initial presence (however, because receiving the roster may not be desirable for all resources, e.g., a connection with limited bandwidth, the client's request for the roster is OPTIONAL). If an available resource does not request the roster during a session, the server MUST NOT send it presence subscriptions and associated roster updates.

Example: Client requests current roster from server:

```
<iq from='juliet@example.com/balcony' type='get' id='roster_1'>
  <query xmlns='jabber:iq:roster'/>
</iq>
```

Example: Client receives roster from server:

```
<iq to='juliet@example.com/balcony' type='result' id='roster_1'>
  <query xmlns='jabber:iq:roster'>
    <item jid='romeo@example.net'
          name='Romeo'
          subscription='both'>
      <group>Friends</group>
    </item>
    <item jid='mercutio@example.org'
          name='Mercutio'
          subscription='from'>
      <group>Friends</group>
    </item>
    <item jid='benvolio@example.org'
          name='Benvolio'
          subscription='both'>
      <group>Friends</group>
    </item>
  </query>
</iq>
```

## 7.4. Adding a Roster Item

TOC

At any time, a user MAY add an item to his or her roster.

Example: Client adds a new item:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 526
Keysight, EX-1045
Page 526

```
<iq from='juliet@example.com/balcony' type='set' id='roster_2'>
  <query xmlns='jabber:iq:roster'>
    <item jid='nurse@example.com'
          name='Nurse'>
      <group>Servants</group>
    </item>
  </query>
</iq>
```

The server MUST update the roster information in persistent storage, and also push the change out to all of the user's available resources that have requested the roster. This "roster push" consists of an IQ stanza of type "set" from the server to the client and enables all available resources to remain in sync with the server-based roster information.

Example: Server (1) pushes the updated roster information to all available resources that have requested the roster and (2) replies with an IQ result to the sending resource:

```
<iq to='juliet@example.com/balcony'
    type='set'
    id='a78b4q6ha463'>
  <query xmlns='jabber:iq:roster'>
    <item jid='nurse@example.com'
          name='Nurse'
          subscription='none'>
      <group>Servants</group>
    </item>
  </query>
</iq>

<iq to='juliet@example.com/chamber'
    type='set'
    id='a78b4q6ha464'>
  <query xmlns='jabber:iq:roster'>
    <item jid='nurse@example.com'
          name='Nurse'
          subscription='none'>
      <group>Servants</group>
    </item>
  </query>
</iq>

<iq to='juliet@example.com/balcony' type='result' id='roster_2'/>
```

As required by the semantics of the IQ stanza kind as defined in **[XMPP-CORE]**, each resource that received the roster push MUST reply with an IQ stanza of type "result" (or "error").

Example: Resources reply with an IQ result to the server:

```
<iq from='juliet@example.com/balcony'
    to='example.com'
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 527
Keysight, EX-1045
Page 527

```
    type='result'
    id='a78b4q6ha463'/>
<iq from='juliet@example.com/chamber'
    to='example.com'
    type='result'
    id='a78b4q6ha464'/>
```

## 7.5. Updating a Roster Item

TOC

Updating an existing roster item (e.g., changing the group) is done in the same way as adding a new roster item, i.e., by sending the roster item in an IQ set to the server.

Example: User updates roster item (added group):

```
<iq from='juliet@example.com/chamber' type='set' id='roster_3'>
  <query xmlns='jabber:iq:roster'>
    <item jid='romeo@example.net'
          name='Romeo'
          subscription='both'>
      <group>Friends</group>
      <group>Lovers</group>
    </item>
  </query>
</iq>
```

As with adding a roster item, when updating a roster item the server MUST update the roster information in persistent storage, and also initiate a roster push to all of the user's available resources that have requested the roster.

## 7.6. Deleting a Roster Item

TOC

At any time, a user MAY delete an item from his or her roster by sending an IQ set to the server and making sure that the value of the 'subscription' attribute is "remove" (a compliant server MUST ignore any other values of the 'subscription' attribute when received from a client).

Example: Client removes an item:

```
<iq from='juliet@example.com/balcony' type='set' id='roster_4'>
  <query xmlns='jabber:iq:roster'>
    <item jid='nurse@example.com' subscription='remove'/>
  </query>
</iq>
```

As with adding a roster item, when deleting a roster item the server MUST update the roster information in persistent storage, initiate a roster push to all of the user's available resources that have requested the roster (with the 'subscription' attribute set to a value of "remove"), and send an IQ result to the initiating resource.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 528
Keysight, EX-1045
Page 528

For further information about the implications of this command, see **Removing a Roster Item and Cancelling All Subscriptions**.

## 8. Integration of Roster Items and Presence Subscriptions

TOC

### 8.1. Overview

TOC

Some level of integration between roster items and presence subscriptions is normally expected by an instant messaging user regarding the user's subscriptions to and from other contacts. This section describes the level of integration that MUST be supported within XMPP instant messaging applications.

There are four primary subscription states:

- None -- the user does not have a subscription to the contact's presence information, and the contact does not have a subscription to the user's presence information
- To -- the user has a subscription to the contact's presence information, but the contact does not have a subscription to the user's presence information
- From -- the contact has a subscription to the user's presence information, but the user does not have a subscription to the contact's presence information
- Both -- both the user and the contact have subscriptions to each other's presence information (i.e., the union of 'from' and 'to')

Each of these states is reflected in the roster of both the user and the contact, thus resulting in durable subscription states. Narrative explanations of how these subscription states interact with roster items in order to complete certain defined use cases are provided in the following sub-sections. Full details regarding server and client handling of all subscription states (including pending states between the primary states listed above) is provided in **Subscription States**.

The server MUST NOT send presence subscription requests or roster pushes to unavailable resources, nor to available resources that have not requested the roster.

The 'from' and 'to' addresses are OPTIONAL in roster pushes; if included, their values SHOULD be the full JID of the resource for that session. A client MUST acknowledge each roster push with an IQ stanza of type "result" (for the sake of brevity, these stanzas are not shown in the following examples but are required by the IQ semantics defined in **[XMPP-CORE]**).

### 8.2. User Subscribes to Contact

TOC

The process by which a user subscribes to a contact, including the interaction between roster items and subscription states, is described below.

1. In preparation for being able to render the contact in the user's client interface and for the server to keep track of the subscription, the user's



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 529
Keysight, EX-1045
Page 529

client SHOULD perform a "roster set" for the new roster item. This request consists of sending an IQ stanza of type='set' containing a <query/> element qualified by the 'jabber:iq:roster' namespace, which in turn contains an <item/> element that defines the new roster item; the <item/> element MUST possess a 'jid' attribute, MAY possess a 'name' attribute, MUST NOT possess a 'subscription' attribute, and MAY contain one or more <group/> child elements:

```
<iq type='set' id='set1'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        name='MyContact'>
      <group>MyBuddies</group>
    </item>
  </query>
</iq>
```

2. As a result, the user's server (1) MUST initiate a roster push for the new roster item to all available resources associated with this user that have requested the roster, setting the 'subscription' attribute to a value of "none"; and (2) MUST reply to the sending resource with an IQ result indicating the success of the roster set:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        subscription='none'
        name='MyContact'>
      <group>MyBuddies</group>
    </item>
  </query>
</iq>

<iq type='result' id='set1'/>
```

3. If the user wants to request a subscription to the contact's presence information, the user's client MUST send a presence stanza of type='subscribe' to the contact:

```
<presence to='contact@example.org' type='subscribe'/>
```

4. As a result, the user's server MUST initiate a second roster push to all of the user's available resources that have requested the roster, setting the contact to the pending sub-state of the 'none' subscription state; this pending sub-state is denoted by the inclusion of the ask='subscribe' attribute in the roster item:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        subscription='none'
        ask='subscribe'
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 530
Keysight, EX-1045
Page 530

```
        name='MyContact'>
      <group>MyBuddies</group>
    </item>
  </query>
</iq>
```

Note: If the user did not create a roster item before sending the subscription request, the server MUST now create one on behalf of the user, then send a roster push to all of the user's available resources that have requested the roster, absent the 'name' attribute and the <group/> child shown above.

5. The user's server MUST also stamp the presence stanza of type "subscribe" with the user's bare JID (i.e., <user@example.com>) as the 'from' address (if the user provided a 'from' address set to the user's full JID, the server SHOULD remove the resource identifier). If the contact is served by a different host than the user, the user's server MUST route the presence stanza to the contact's server for delivery to the contact (this case is assumed throughout; however, if the contact is served by the same host, then the server can simply deliver the presence stanza directly):

```
<presence
    from='user@example.com'
    to='contact@example.org'
    type='subscribe'/>
```

Note: If the user's server receives a presence stanza of type "error" from the contact's server, it MUST deliver the error stanza to the user, whose client MAY determine that the error is in response to the outgoing presence stanza of type "subscribe" it sent previously (e.g., by tracking an 'id' attribute) and then choose to resend the "subscribe" request or revert the roster to its previous state by sending a presence stanza of type "unsubscribe" to the contact.

6. Upon receiving the presence stanza of type "subscribe" addressed to the contact, the contact's server MUST determine if there is at least one available resource from which the contact has requested the roster. If so, it MUST deliver the subscription request to the contact (if not, the contact's server MUST store the subscription request offline for delivery when this condition is next met; normally this is done by adding a roster item for the contact to the user's roster, with a state of "None + Pending In" as defined under **Subscription States**, however a server SHOULD NOT push or deliver roster items in that state to the contact). No matter when the subscription request is delivered, the contact must decide whether or not to approve it (subject to the contact's configured preferences, the contact's client MAY approve or refuse the subscription request without presenting it to the contact). Here we assume the "happy path" that the contact approves the subscription request (the alternate flow of declining the subscription request is defined in **Section 8.2.1**). In this case, the contact's client (1) SHOULD perform a roster set specifying the desired nickname and group for the user (if any); and (2) MUST send a presence stanza of type "subscribed" to the user in order to approve the subscription request.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 531
Keysight, EX-1045
Page 531

```
<iq type='set' id='set2'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
  </query>
</iq>

<presence to='user@example.com' type='subscribed'/>
```

7. As a result, the contact's server (1) MUST initiate a roster push to all available resources associated with the contact that have requested the roster, containing a roster item for the user with the subscription state set to 'from' (the server MUST send this even if the contact did not perform a roster set); (2) MUST return an IQ result to the sending resource indicating the success of the roster set; (3) MUST route the presence stanza of type "subscribed" to the user, first stamping the 'from' address as the bare JID (<contact@example.org>) of the contact; and (4) MUST send available presence from all of the contact's available resources to the user:

```
<iq type='set' to='contact@example.org/resource'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        subscription='from'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
  </query>
</iq>

<iq type='result' to='contact@example.org/resource' id='set2'/>

<presence
    from='contact@example.org'
    to='user@example.com'
    type='subscribed'/>

<presence
    from='contact@example.org/resource'
    to='user@example.com'/>
```

Note: If the contact's server receives a presence stanza of type "error" from the user's server, it MUST deliver the error stanza to the contact, whose client MAY determine that the error is in response to the outgoing presence stanza of type "subscribed" it sent previously (e.g., by tracking an 'id' attribute) and then choose to resend the "subscribed" notification or revert the roster to its previous state by sending a presence stanza of type "unsubscribed" to the user.

8. Upon receiving the presence stanza of type "subscribed" addressed to the



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 532
Keysight, EX-1045
Page 532

user, the user's server MUST first verify that the contact is in the user's roster with either of the following states: (a) subscription='none' and ask='subscribe' or (b) subscription='from' and ask='subscribe'. If the contact is not in the user's roster with either of those states, the user's server MUST silently ignore the presence stanza of type "subscribed" (i.e., it MUST NOT route it to the user, modify the user's roster, or generate a roster push to the user's available resources). If the contact is in the user's roster with either of those states, the user's server (1) MUST deliver the presence stanza of type "subscribed" from the contact to the user; (2) MUST initiate a roster push to all of the user's available resources that have requested the roster, containing an updated roster item for the contact with the 'subscription' attribute set to a value of "to"; and (3) MUST deliver the available presence stanza received from each of the contact's available resources to each of the user's available resources:

```
<presence
    to='user@example.com'
    from='contact@example.org'
    type='subscribed'/>

<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        subscription='to'
        name='MyContact'>
      <group>MyBuddies</group>
    </item>
  </query>
</iq>

<presence
    from='contact@example.org/resource'
    to='user@example.com/resource'/>
```

9. Upon receiving the presence stanza of type "subscribed", the user SHOULD acknowledge receipt of that subscription state notification through either "affirming" it by sending a presence stanza of type "subscribe" to the contact or "denying" it by sending a presence stanza of type "unsubscribe" to the contact; this step does not necessarily affect the subscription state (see **Subscription States** for details), but instead lets the user's server know that it MUST no longer send notification of the subscription state change to the user (see **Section 9.4**).

From the perspective of the user, there now exists a subscription to the contact's presence information; from the perspective of the contact, there now exists a subscription from the user.

### 8.2.1. Alternate Flow: Contact Declines Subscription Request

TOC

The above activity flow represents the "happy path" regarding the user's subscription request to the contact. The main alternate flow occurs if the contact refuses the user's subscription request, as described below.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 533
Keysight, EX-1045
Page 533

1. If the contact wants to refuse the request, the contact's client MUST send a presence stanza of type "unsubscribed" to the user (instead of the presence stanza of type "subscribed" sent in Step 6 of **Section 8.2**):

```
<presence to='user@example.com' type='unsubscribed'/>
```

2. As a result, the contact's server MUST route the presence stanza of type "unsubscribed" to the user, first stamping the 'from' address as the bare JID (<contact@example.org>) of the contact:

```
<presence
    from='contact@example.org'
    to='user@example.com'
    type='unsubscribed'/>
```

   Note: If the contact's server previously added the user to the contact's roster for tracking purposes, it MUST remove the relevant item at this time.

3. Upon receiving the presence stanza of type "unsubscribed" addressed to the user, the user's server (1) MUST deliver that presence stanza to the user and (2) MUST initiate a roster push to all of the user's available resources that have requested the roster, containing an updated roster item for the contact with the 'subscription' attribute set to a value of "none" and with no 'ask' attribute:

```
<presence
    from='contact@example.org'
    to='user@example.com'
    type='unsubscribed'/>

<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        subscription='none'
        name='MyContact'>
      <group>MyBuddies</group>
    </item>
  </query>
</iq>
```

4. Upon receiving the presence stanza of type "unsubscribed", the user SHOULD acknowledge receipt of that subscription state notification through either "affirming" it by sending a presence stanza of type "unsubscribe" to the contact or "denying" it by sending a presence stanza of type "subscribe" to the contact; this step does not necessarily affect the subscription state (see **Subscription States** for details), but instead lets the user's server know that it MUST no longer send notification of the subscription state change to the user (see **Section 9.4**).

As a result of this activity, the contact is now in the user's roster with a subscription state of "none", whereas the user is not in the contact's roster at all.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 534
Keysight, EX-1045
Page 534

## 8.3. Creating a Mutual Subscription

TOC

The user and contact can build on the "happy path" described above to create a mutual subscription (i.e., a subscription of type "both"). The process is described below.

1. If the contact wants to create a mutual subscription, the contact MUST send a subscription request to the user (subject to the contact's configured preferences, the contact's client MAY send this automatically):

```
<presence to='user@example.com' type='subscribe'/>
```

2. As a result, the contact's server (1) MUST initiate a roster push to all available resources associated with the contact that have requested the roster, with the user still in the 'from' subscription state but with a pending 'to' subscription denoted by the inclusion of the ask='subscribe' attribute in the roster item; and (2) MUST route the presence stanza of type "subscribe" to the user, first stamping the 'from' address as the bare JID (<contact@example.org>) of the contact:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        subscription='from'
        ask='subscribe'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
  </query>
</iq>

<presence
    from='contact@example.org'
    to='user@example.com'
    type='subscribe'/>
```

   Note: If the contact's server receives a presence stanza of type "error" from the user's server, it MUST deliver the error stanza to the contact, whose client MAY determine that the error is in response to the outgoing presence stanza of type "subscribe" it sent previously (e.g., by tracking an 'id' attribute) and then choose to resend the "subscribe" request or revert the roster to its previous state by sending a presence stanza of type "unsubscribe" to the user.

3. Upon receiving the presence stanza of type "subscribe" addressed to the user, the user's server must determine if there is at least one available resource for which the user has requested the roster. If so, the user's server MUST deliver the subscription request to the user (if not, it MUST store the subscription request offline for delivery when this condition is next met). No matter when the subscription request is delivered, the user must then decide whether or not to approve it (subject to the user's configured preferences, the user's client MAY approve or refuse the subscription



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 535
Keysight, EX-1045
Page 535

request without presenting it to the user). Here we assume the "happy path" that the user approves the subscription request (the alternate flow of declining the subscription request is defined in **Section 8.3.1**). In this case, the user's client MUST send a presence stanza of type "subscribed" to the contact in order to approve the subscription request.

```
<presence to='contact@example.org' type='subscribed'/>
```

4. As a result, the user's server (1) MUST initiate a roster push to all of the user's available resources that have requested the roster, containing a roster item for the contact with the 'subscription' attribute set to a value of "both"; (2) MUST route the presence stanza of type "subscribed" to the contact, first stamping the 'from' address as the bare JID (<user@example.com>) of the user; and (3) MUST send to the contact the full XML of the last presence stanza with no 'to' attribute received by the server from each of the user's available resources (subject to privacy lists in force for each session):

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        subscription='both'
        name='MyContact'>
      <group>MyBuddies</group>
    </item>
  </query>
</iq>

<presence
    from='user@example.com'
    to='contact@example.org'
    type='subscribed'/>

<presence
    from='user@example.com/resource'
    to='contact@example.org'/>
```

   Note: If the user's server receives a presence stanza of type "error" from the contact's server, it MUST deliver the error stanza to the user, whose client MAY determine that the error is in response to the outgoing presence stanza of type "subscribed" it sent previously (e.g., by tracking an 'id' attribute) and then choose to resend the subscription request or revert the roster to its previous state by sending a presence stanza of type "unsubscribed" to the contact.

5. Upon receiving the presence stanza of type "subscribed" addressed to the contact, the contact's server MUST first verify that the user is in the contact's roster with either of the following states: (a) subscription='none' and ask='subscribe' or (b) subscription='from' and ask='subscribe'. If the user is not in the contact's roster with either of those states, the contact's server MUST silently ignore the presence stanza of type "subscribed" (i.e., it MUST NOT route it to the contact, modify the contact's roster, or generate a roster push to the contact's available resources). If the user is in the



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 536
Keysight, EX-1045
Page 536

contact's roster with either of those states, the contact's server (1) MUST deliver the presence stanza of type "subscribed" from the user to the contact; (2) MUST initiate a roster push to all available resources associated with the contact that have requested the roster, containing an updated roster item for the user with the 'subscription' attribute set to a value of "both"; and (3) MUST deliver the available presence stanza received from each of the user's available resources to each of the contact's available resources:

```
<presence
    from='user@example.com'
    to='contact@example.org'
    type='subscribed'/>

<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        subscription='both'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
  </query>
</iq>

<presence
    from='user@example.com/resource'
    to='contact@example.org/resource'/>
```

6. Upon receiving the presence stanza of type "subscribed", the contact SHOULD acknowledge receipt of that subscription state notification through either "affirming" it by sending a presence stanza of type "subscribe" to the user or "denying" it by sending a presence stanza of type "unsubscribe" to the user; this step does not necessarily affect the subscription state (see **Subscription States** for details), but instead lets the contact's server know that it MUST no longer send notification of the subscription state change to the contact (see **Section 9.4**).

The user and the contact now have a mutual subscription to each other's presence -- i.e., the subscription is of type "both".

---

### 8.3.1. Alternate Flow: User Declines Subscription Request

TOC

The above activity flow represents the "happy path" regarding the contact's subscription request to the user. The main alternate flow occurs if the user refuses the contact's subscription request, as described below.

1. If the user wants to refuse the request, the user's client MUST send a presence stanza of type "unsubscribed" to the contact (instead of the presence stanza of type "subscribed" sent in Step 3 of **Section 8.3**):

```
<presence to='contact@example.org' type='unsubscribed'/>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 537
Keysight, EX-1045
Page 537

2. As a result, the user's server MUST route the presence stanza of type "unsubscribed" to the contact, first stamping the 'from' address as the bare JID (<user@example.com>) of the user:

```
<presence
    from='user@example.com'
    to='contact@example.org'
    type='unsubscribed'/>
```

3. Upon receiving the presence stanza of type "unsubscribed" addressed to the contact, the contact's server (1) MUST deliver that presence stanza to the contact; and (2) MUST initiate a roster push to all available resources associated with the contact that have requested the roster, containing an updated roster item for the user with the 'subscription' attribute set to a value of "from" and with no 'ask' attribute:

```
<presence
    from='user@example.com'
    to='contact@example.org'
    type='unsubscribed'/>

<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        subscription='from'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
  </query>
</iq>
```

4. Upon receiving the presence stanza of type "unsubscribed", the contact SHOULD acknowledge receipt of that subscription state notification through either "affirming" it by sending a presence stanza of type "unsubscribe" to the user or "denying" it by sending a presence stanza of type "subscribe" to the user; this step does not necessarily affect the subscription state (see **Subscription States** for details), but instead lets the contact's server know that it MUST no longer send notification of the subscription state change to the contact (see **Section 9.4**).

As a result of this activity, there has been no change in the subscription state; i.e., the contact is in the user's roster with a subscription state of "to" and the user is in the contact's roster with a subscription state of "from".

---

## 8.4. Unsubscribing

TOC

At any time after subscribing to a contact's presence information, a user MAY unsubscribe. While the XML that the user sends to make this happen is the same in all instances, the subsequent subscription state is different depending on the subscription state obtaining when the unsubscribe "command" is sent. Both possible scenarios are described below.

### 8.4.1. Case #1: Unsubscribing When Subscription is Not Mutual

TOC

In the first case, the user has a subscription to the contact's presence information but the contact does not have a subscription to the user's presence information (i.e., the subscription is not yet mutual).

1. If the user wants to unsubscribe from the contact's presence information, the user MUST send a presence stanza of type "unsubscribe" to the contact:

```
<presence to='contact@example.org' type='unsubscribe'/>
```

2. As a result, the user's server (1) MUST send a roster push to all of the user's available resources that have requested the roster, containing an updated roster item for the contact with the 'subscription' attribute set to a value of "none"; and (2) MUST route the presence stanza of type "unsubscribe" to the contact, first stamping the 'from' address as the bare JID (<user@example.com>) of the user:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        subscription='none'
        name='MyContact'>
      <group>MyBuddies</group>
    </item>
  </query>
</iq>

<presence
    from='user@example.com'
    to='contact@example.org'
    type='unsubscribe'/>
```

3. Upon receiving the presence stanza of type "unsubscribe" addressed to the contact, the contact's server (1) MUST initiate a roster push to all available resources associated with the contact that have requested the roster, containing an updated roster item for the user with the 'subscription' attribute set to a value of "none" (if the contact is unavailable or has not requested the roster, the contact's server MUST modify the roster item and send that modified item the next time the contact requests the roster); and (2) MUST deliver the "unsubscribe" state change notification to the contact:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        subscription='none'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
  </query>
</iq>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 539
Keysight, EX-1045
Page 539

```
<presence
    from='user@example.com'
    to='contact@example.org'
    type='unsubscribe'/>
```

4. Upon receiving the presence stanza of type "unsubscribe", the contact SHOULD acknowledge receipt of that subscription state notification through either "affirming" it by sending a presence stanza of type "unsubscribed" to the user or "denying" it by sending a presence stanza of type "subscribed" to the user; this step does not necessarily affect the subscription state (see **Subscription States** for details), but instead lets the contact's server know that it MUST no longer send notification of the subscription state change to the contact (see **Section 9.4**).
5. The contact's server then (1) MUST send a presence stanza of type "unsubscribed" to the user; and (2) SHOULD send unavailable presence from all of the contact's available resources to the user:

```
<presence
    from='contact@example.org'
    to='user@example.com'
    type='unsubscribed'/>

<presence
    from='contact@example.org/resource'
    to='user@example.com'
    type='unavailable'/>
```

6. When the user's server receives the presence stanzas of type "unsubscribed" and "unavailable", it MUST deliver them to the user:

```
<presence
    from='contact@example.org'
    to='user@example.com'
    type='unsubscribed'/>

<presence
    from='contact@example.org/resource'
    to='user@example.com'
    type='unavailable'/>
```

7. Upon receiving the presence stanza of type "unsubscribed", the user SHOULD acknowledge receipt of that subscription state notification through either "affirming" it by sending a presence stanza of type "unsubscribe" to the contact or "denying" it by sending a presence stanza of type "subscribe" to the contact; this step does not necessarily affect the subscription state (see **Subscription States** for details), but instead lets the user's server know that it MUST no longer send notification of the subscription state change to the user (see **Section 9.4**).

---

### 8.4.2. Case #2: Unsubscribing When Subscription is Mutual

TOC

In the second case, the user has a subscription to the contact's presence information



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 540
Keysight, EX-1045
Page 540

and the contact also has a subscription to the user's presence information (i.e., the subscription is mutual).

1. If the user wants to unsubscribe from the contact's presence information, the user MUST send a presence stanza of type "unsubscribe" to the contact:

```
<presence to='contact@example.org' type='unsubscribe'/>
```

2. As a result, the user's server (1) MUST send a roster push to all of the user's available resources that have requested the roster, containing an updated roster item for the contact with the 'subscription' attribute set to a value of "from"; and (2) MUST route the presence stanza of type "unsubscribe" to the contact, first stamping the 'from' address as the bare JID (<user@example.com>) of the user:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        subscription='from'
        name='MyContact'>
      <group>MyBuddies</group>
    </item>
  </query>
</iq>

<presence
    from='user@example.com'
    to='contact@example.org'
    type='unsubscribe'/>
```

3. Upon receiving the presence stanza of type "unsubscribe" addressed to the contact, the contact's server (1) MUST initiate a roster push to all available resources associated with the contact that have requested the roster, containing an updated roster item for the user with the 'subscription' attribute set to a value of "to" (if the contact is unavailable or has not requested the roster, the contact's server MUST modify the roster item and send that modified item the next time the contact requests the roster); and (2) MUST deliver the "unsubscribe" state change notification to the contact:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        subscription='to'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
  </query>
</iq>

<presence
    from='user@example.com'
    to='contact@example.org'
    type='unsubscribe'/>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 541
Keysight, EX-1045
Page 541

4. Upon receiving the presence stanza of type "unsubscribe", the contact SHOULD acknowledge receipt of that subscription state notification through either "affirming" it by sending a presence stanza of type "unsubscribed" to the user or "denying" it by sending a presence stanza of type "subscribed" to the user; this step does not necessarily affect the subscription state (see **Subscription States** for details), but instead lets the contact's server know that it MUST no longer send notification of the subscription state change to the contact (see **Section 9.4**).
5. The contact's server then (1) MUST send a presence stanza of type "unsubscribed" to the user; and (2) SHOULD send unavailable presence from all of the contact's available resources to the user:

```
<presence
    from='contact@example.org'
    to='user@example.com'
    type='unsubscribed'/>

<presence
    from='contact@example.org/resource'
    to='user@example.com'
    type='unavailable'/>
```

6. When the user's server receives the presence stanzas of type "unsubscribed" and "unavailable", it MUST deliver them to the user:

```
<presence
    from='contact@example.org'
    to='user@example.com'
    type='unsubscribed'/>

<presence
    from='contact@example.org/resource'
    to='user@example.com'
    type='unavailable'/>
```

7. Upon receiving the presence stanza of type "unsubscribed", the user SHOULD acknowledge receipt of that subscription state notification through either "affirming" it by sending a presence stanza of type "unsubscribe" to the contact or "denying" it by sending a presence stanza of type "subscribe" to the contact; this step does not necessarily affect the subscription state (see **Subscription States** for details), but instead lets the user's server know that it MUST no longer send notification of the subscription state change to the user (see **Section 9.4**).

Note: Obviously this does not result in removal of the roster item from the user's roster, and the contact still has a subscription to the user's presence information. In order to both completely cancel a mutual subscription and fully remove the roster item from the user's roster, the user SHOULD update the roster item with subscription='remove' as defined under **Removing a Roster Item and Cancelling All Subscriptions**.

---

### 8.5. Cancelling a Subscription

TOC

At any time after approving a subscription request from a user, a contact MAY cancel



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 542
Keysight, EX-1045
Page 542

that subscription. While the XML that the contact sends to make this happen is the same in all instances, the subsequent subscription state is different depending on the subscription state obtaining when the cancellation was sent. Both possible scenarios are described below.

### 8.5.1. Case #1: Cancelling When Subscription is Not Mutual

TOC

In the first case, the user has a subscription to the contact's presence information but the contact does not have a subscription to the user's presence information (i.e., the subscription is not yet mutual).

1. If the contact wants to cancel the user's subscription, the contact MUST send a presence stanza of type "unsubscribed" to the user:

```
<presence to='user@example.com' type='unsubscribed'/>
```

2. As a result, the contact's server (1) MUST send a roster push to all of the contact's available resources that have requested the roster, containing an updated roster item for the user with the 'subscription' attribute set to a value of "none"; (2) MUST route the presence stanza of type "unsubscribed" to the user, first stamping the 'from' address as the bare JID (<contact@example.org>) of the contact; and (3) SHOULD send unavailable presence from all of the contact's available resources to the user:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        subscription='none'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
  </query>
</iq>

<presence
    from='contact@example.org'
    to='user@example.com'
    type='unsubscribed'/>

<presence
    from='contact@example.org/resource'
    to='user@example.com'
    type='unavailable'/>
```

3. Upon receiving the presence stanza of type "unsubscribed" addressed to the user, the user's server (1) MUST initiate a roster push to all of the user's available resources that have requested the roster, containing an updated roster item for the contact with the 'subscription' attribute set to a value of "none" (if the user is unavailable or has not requested the roster, the user's server MUST modify the roster item and send that modified item the next time the user requests the roster); (2) MUST deliver the "unsubscribed" state change notification to all of the user's available resources; and (3)



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 543
Keysight, EX-1045
Page 543

MUST deliver the unavailable presence to all of the user's available resources:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        subscription='none'
        name='MyContact'>
      <group>MyBuddies</group>
    </item>
  </query>
</iq>

<presence
    from='contact@example.org'
    to='user@example.com'
    type='unsubscribed'/>

<presence
    from='contact@example.org/resource'
    to='user@example.com'
    type='unavailable'/>
```

4. Upon receiving the presence stanza of type "unsubscribed", the user SHOULD acknowledge receipt of that subscription state notification through either "affirming" it by sending a presence stanza of type "unsubscribe" to the contact or "denying" it by sending a presence stanza of type "subscribe" to the contact; this step does not necessarily affect the subscription state (see **Subscription States** for details), but instead lets the user's server know that it MUST no longer send notification of the subscription state change to the user (see **Section 9.4**).

---

### 8.5.2. Case #2: Cancelling When Subscription is Mutual

TOC

In the second case, the user has a subscription to the contact's presence information and the contact also has a subscription to the user's presence information (i.e., the subscription is mutual).

1. If the contact wants to cancel the user's subscription, the contact MUST send a presence stanza of type "unsubscribed" to the user:

```
<presence to='user@example.com' type='unsubscribed'/>
```

2. As a result, the contact's server (1) MUST send a roster push to all of the contact's available resources that have requested the roster, containing an updated roster item for the user with the 'subscription' attribute set to a value of "to"; (2) MUST route the presence stanza of type "unsubscribed" to the user, first stamping the 'from' address as the bare JID (<contact@example.org>) of the contact; and (3) SHOULD send unavailable presence from all of the contact's available resources to all of the user's available resources:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 544
Keysight, EX-1045
Page 544

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        subscription='to'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
  </query>
</iq>

<presence
    from='contact@example.org'
    to='user@example.com'
    type='unsubscribed'/>

<presence
    from='contact@example.org/resource'
    to='user@example.com'
    type='unavailable'/>
```

3. Upon receiving the presence stanza of type "unsubscribed" addressed to the user, the user's server (1) MUST initiate a roster push to all of the user's available resources that have requested the roster, containing an updated roster item for the contact with the 'subscription' attribute set to a value of "from" (if the user is unavailable or has not requested the roster, the user's server MUST modify the roster item and send that modified item the next time the user requests the roster); and (2) MUST deliver the "unsubscribed" state change notification to all of the user's available resources; and (3) MUST deliver the unavailable presence to all of the user's available resources:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        subscription='from'
        name='MyContact'>
      <group>MyBuddies</group>
    </item>
  </query>
</iq>

<presence
    from='contact@example.org'
    to='user@example.com'
    type='unsubscribed'/>

<presence
    from='contact@example.org/resource'
    to='user@example.com'
    type='unavailable'/>
```

4. Upon receiving the presence stanza of type "unsubscribed", the user SHOULD acknowledge receipt of that subscription state notification through



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 545
Keysight, EX-1045
Page 545

either "affirming" it by sending a presence stanza of type "unsubscribe" to the contact or "denying" it by sending a presence stanza of type "subscribe" to the contact; this step does not necessarily affect the subscription state (see **Subscription States** for details), but instead lets the user's server know that it MUST no longer send notification of the subscription state change to the user (see **Section 9.4**).

Note: Obviously this does not result in removal of the roster item from the contact's roster, and the contact still has a subscription to the user's presence information. In order to both completely cancel a mutual subscription and fully remove the roster item from the contact's roster, the contact should update the roster item with subscription='remove' as defined under **Removing a Roster Item and Cancelling All Subscriptions**.

---

## 8.6. Removing a Roster Item and Cancelling All Subscriptions

TOC

Because there may be many steps involved in completely removing a roster item and cancelling subscriptions in both directions, the roster management protocol includes a "shortcut" method for doing so. The process may be initiated no matter what the current subscription state is by sending a roster set containing an item for the contact with the 'subscription' attribute set to a value of "remove":

```
<iq type='set' id='remove1'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='contact@example.org'
        subscription='remove'/>
  </query>
</iq>
```

When the user removes a contact from his or her roster by setting the 'subscription' attribute to a value of "remove", the user's server (1) MUST automatically cancel any existing presence subscription between the user and the contact (both 'to' and 'from' as appropriate); (2) MUST remove the roster item from the user's roster and inform all of the user's available resources that have requested the roster of the roster item removal; (3) MUST inform the resource that initiated the removal of success; and (4) SHOULD send unavailable presence from all of the user's available resources to the contact:

```
<presence
    from='user@example.com'
    to='contact@example.org'
    type='unsubscribe'/>

<presence
    from='user@example.com'
    to='contact@example.org'
    type='unsubscribed'/>

<iq type='set'>
  <query xmlns='jabber:iq:roster'>
```

```
    <item
        jid='contact@example.org'
        subscription='remove'/>
  </query>
</iq>

<iq type='result' id='remove1'/>

<presence
    from='user@example.com/resource'
    to='contact@example.org'
    type='unavailable'/>
```

Upon receiving the presence stanza of type "unsubscribe", the contact's server (1) MUST initiate a roster push to all available resources associated with the contact that have requested the roster, containing an updated roster item for the user with the 'subscription' attribute set to a value of "to" (if the contact is unavailable or has not requested the roster, the contact's server MUST modify the roster item and send that modified item the next time the contact requests the roster); and (2) MUST also deliver the "unsubscribe" state change notification to all of the contact's available resources:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        subscription='to'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
  </query>
</iq>

<presence
    from='user@example.com'
    to='contact@example.org'
    type='unsubscribe'/>
```

Upon receiving the presence stanza of type "unsubscribed", the contact's server (1) MUST initiate a roster push to all available resources associated with the contact that have requested the roster, containing an updated roster item for the user with the 'subscription' attribute set to a value of "none" (if the contact is unavailable or has not requested the roster, the contact's server MUST modify the roster item and send that modified item the next time the contact requests the roster); and (2) MUST also deliver the "unsubscribe" state change notification to all of the contact's available resources:

```
<iq type='set'>
  <query xmlns='jabber:iq:roster'>
    <item
        jid='user@example.com'
        subscription='none'
        name='SomeUser'>
      <group>SomeGroup</group>
    </item>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 547
Keysight, EX-1045
Page 547

```
  </query>
</iq>

<presence
    from='user@example.com'
    to='contact@example.org'
    type='unsubscribed'/>
```

Upon receiving the presence stanza of type "unavailable" addressed to the contact, the contact's server MUST deliver the unavailable presence to all of the user's available resources:

```
<presence
    from='user@example.com/resource'
    to='contact@example.org'
    type='unavailable'/>
```

Note: When the user removes the contact from the user's roster, the end state of the contact's roster is that the user is still in the contact's roster with a subscription state of "none"; in order to completely remove the roster item for the user, the contact needs to also send a roster removal request.

## 9. Subscription States

TOC

This section provides detailed information about subscription states and server handling of subscription-related presence stanzas (i.e., presence stanzas of type "subscribe", "subscribed", "unsubscribe", and "unsubscribed").

### 9.1. Defined States

TOC

There are nine possible subscription states, which are described here from the user's (not contact's) perspective:

1. "None" = contact and user are not subscribed to each other, and neither has requested a subscription from the other
2. "None + Pending Out" = contact and user are not subscribed to each other, and user has sent contact a subscription request but contact has not replied yet
3. "None + Pending In" = contact and user are not subscribed to each other, and contact has sent user a subscription request but user has not replied yet (note: contact's server SHOULD NOT push or deliver roster items in this state, but instead SHOULD wait until contact has approved subscription request from user)
4. "None + Pending Out/In" = contact and user are not subscribed to each other, contact has sent user a subscription request but user has not replied yet, and user has sent contact a subscription request but contact has not replied yet
5. "To" = user is subscribed to contact (one-way)



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 548
Keysight, EX-1045
Page 548

6. "To + Pending In" = user is subscribed to contact, and contact has sent user a subscription request but user has not replied yet
7. "From" = contact is subscribed to user (one-way)
8. "From + Pending Out" = contact is subscribed to user, and user has sent contact a subscription request but contact has not replied yet
9. "Both" = user and contact are subscribed to each other (two-way)

## 9.2. Server Handling of Outbound Presence Subscription Stanzas

TOC

Outbound presence subscription stanzas enable the user to manage his or her subscription to the contact's presence information (via the "subscribe" and "unsubscribe" types), and to manage the contact's access to the user's presence information (via the "subscribed" and "unsubscribed" types).

Because it is possible for the user's server and the contact's server to lose synchronization regarding subscription states, the user's server MUST without exception route all outbound presence stanzas of type "subscribe" or "unsubscribe" to the contact so that the user is able to resynchronize his or her subscription to the contact's presence information if needed.

The user's server SHOULD NOT route a presence stanza of type "subscribed" or "unsubscribed" to the contact if the stanza does not result in a subscription state change from the user's perspective, and MUST NOT make a state change. If the stanza results in a subscription state change, the user's server MUST route the stanza to the contact and MUST make the appropriate state change. These rules are summarized in the following tables.

Table 1: Recommended handling of outbound "subscribed" stanzas

| EXISTING STATE | ROUTE? | NEW STATE |
|---|---|---|
| "None" | no | no state change |
| "None + Pending Out" | no | no state change |
| "None + Pending In" | yes | "From" |
| "None + Pending Out/In" | yes | "From + Pending Out" |
| "To" | no | no state change |
| "To + Pending In" | yes | "Both" |
| "From" | no | no state change |
| "From + Pending Out" | no | no state change |
| "Both" | no | no state change |

Table 2: Recommended handling of outbound "unsubscribed" stanzas

| EXISTING STATE | ROUTE? | NEW STATE |
|---|---|---|
| "None" | no | no state change |
| "None + Pending Out" | no | no state change |
| "None + Pending In" | yes | "None" |



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 549
Keysight, EX-1045
Page 549

| "None + Pending Out/In" | yes | "None + Pending Out" |
| "To" | no | no state change |
| "To + Pending In" | yes | "To" |
| "From" | yes | "None" |
| "From + Pending Out" | yes | "None + Pending Out" |
| "Both" | yes | "To" |

## 9.3. Server Handling of Inbound Presence Subscription Stanzas

TOC

Inbound presence subscription stanzas request a subscription-related action from the user (via the "subscribe" type), inform the user of subscription-related actions taken by the contact (via the "unsubscribe" type), or enable the contact to manage the user's access to the contact's presence information (via the "subscribed" and "unsubscribed" types).

When the user's server receives a subscription request for the user from the contact (i.e., a presence stanza of type "subscribe"), it MUST deliver that request to the user for approval if the user has not already granted the contact access to the user's presence information and if there is no pending inbound subscription request; however, the user's server SHOULD NOT deliver the new request if there is a pending inbound subscription request, since the previous subscription request will have been recorded. If the user has already granted the contact access to the user's presence information, the user's server SHOULD auto-reply to an inbound presence stanza of type "subscribe" from the contact by sending a presence stanza of type "subscribed" to the contact on behalf of the user; this rule enables the contact to resynchronize the subscription state if needed. These rules are summarized in the following table.

Table 3: Recommended handling of inbound "subscribe" stanzas

| EXISTING STATE | DELIVER? | NEW STATE |
| "None" | yes | "None + Pending In" |
| "None + Pending Out" | yes | "None + Pending Out/In" |
| "None + Pending In" | no | no state change |
| "None + Pending Out/In" | no | no state change |
| "To" | yes | "To + Pending In" |
| "To + Pending In" | no | no state change |
| "From" | no * | no state change |
| "From + Pending Out" | no * | no state change |
| "Both" | no * | no state change |

* Server SHOULD auto-reply with "subscribed" stanza

When the user's server receives a presence stanza of type "unsubscribe" for the user from the contact, if the stanza results in a subscription state change from the user's perspective then the user's server SHOULD auto-reply by sending a presence stanza of type "unsubscribed" to the contact on behalf of the user, MUST deliver the "unsubscribe" stanza to the user, and MUST change the state. If no subscription state change results, the user's server SHOULD NOT deliver the stanza and MUST NOT change the state.

These rules are summarized in the following table.

Table 4: Recommended handling of inbound "unsubscribe" stanzas

| EXISTING STATE | DELIVER? | NEW STATE |
|---|---|---|
| "None" | no | no state change |
| "None + Pending Out" | no | no state change |
| "None + Pending In" | yes * | "None" |
| "None + Pending Out/In" | yes * | "None + Pending Out" |
| "To" | no | no state change |
| "To + Pending In" | yes * | "To" |
| "From" | yes * | "None" |
| "From + Pending Out" | yes * | "None + Pending Out |
| "Both" | yes * | "To" |

* Server SHOULD auto-reply with "unsubscribed" stanza

When the user's server receives a presence stanza of type "subscribed" for the user from the contact, it MUST NOT deliver the stanza to the user and MUST NOT change the subscription state if there is no pending outbound request for access to the contact's presence information. If there is a pending outbound request for access to the contact's presence information and the inbound presence stanza of type "subscribed" results in a subscription state change, the user's server MUST deliver the stanza to the user and MUST change the subscription state. If the user already has access to the contact's presence information, the inbound presence stanza of type "subscribed" does not result in a subscription state change; therefore the user's server SHOULD NOT deliver the stanza to the user and MUST NOT change the subscription state. These rules are summarized in the following table.

Table 5: Recommended handling of inbound "subscribed" stanzas

| EXISTING STATE | DELIVER? | NEW STATE |
|---|---|---|
| "None" | no | no state change |
| "None + Pending Out" | yes | "To" |
| "None + Pending In" | no | no state change |
| "None + Pending Out/In" | yes | "To + Pending In" |
| "To" | no | no state change |
| "To + Pending In" | no | no state change |
| "From" | no | no state change |
| "From + Pending Out" | yes | "Both" |
| "Both" | no | no state change |

When the user's server receives a presence stanza of type "unsubscribed" for the user from the contact, it MUST deliver the stanza to the user and MUST change the subscription state if there is a pending outbound request for access to the contact's presence information or if the user currently has access to the contact's presence information. Otherwise, the user's server SHOULD NOT deliver the stanza and MUST



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 551
Keysight, EX-1045
Page 551

NOT change the subscription state. These rules are summarized in the following table.

Table 6: Recommended handling of inbound "unsubscribed" stanzas

| EXISTING STATE | DELIVER? | NEW STATE |
|---|---|---|
| "None" | no | no state change |
| "None + Pending Out" | yes | "None" |
| "None + Pending In" | no | no state change |
| "None + Pending Out/In" | yes | "None + Pending In" |
| "To" | yes | "None" |
| "To + Pending In" | yes | "None + Pending In" |
| "From" | no | no state change |
| "From + Pending Out" | yes | "From" |
| "Both" | yes | "From" |

## 9.4. Server Delivery and Client Acknowledgement of Subscription Requests and State Change Notifications

TOC

When a server receives an inbound presence stanza of type "subscribe" (i.e., a subscription request) or of type "subscribed", "unsubscribe", or "unsubscribed" (i.e., a subscription state change notification), in addition to sending the appropriate roster push (or updated roster when the roster is next requested by an available resource), it MUST deliver the request or notification to the intended recipient at least once. A server MAY require the recipient to acknowledge receipt of all state change notifications (and MUST require acknowledgement in the case of subscription requests, i.e., presence stanzas of type "subscribe"). In order to require acknowledgement, a server SHOULD send the request or notification to the recipient each time the recipient logs in, until the recipient acknowledges receipt of the notification by "affirming" or "denying" the notification, as shown in the following table:

Table 7: Acknowledgement of subscription state change notifications

| STANZA TYPE | ACCEPT | DENY |
|---|---|---|
| subscribe | subscribed | unsubscribed |
| subscribed | subscribe | unsubscribe |
| unsubscribe | unsubscribed | subscribed |
| unsubscribed | unsubscribe | subscribe |

Obviously, given the foregoing subscription state charts, some of the acknowledgement stanzas will be routed to the contact and result in subscription state changes, while others will not. However, any such stanzas MUST result in the server's no longer sending the subscription state notification to the user.

Because a user's server MUST automatically generate outbound presence stanzas of type "unsubscribe" and "unsubscribed" upon receiving a roster set with the 'subscription'



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 552
Keysight, EX-1045
Page 552

attribute set to a value of "remove" (see **Removing a Roster Item and Cancelling All Subscriptions**), the server MUST treat a roster remove request as equivalent to sending both of those presence stanzas for purposes of determining whether to continue sending subscription state change notifications of type "subscribe" or "subscribed" to the user.

## 10. Blocking Communication

TOC

Most instant messaging systems have found it necessary to implement some method for users to block communications from particular other users (this is also required by sections 5.1.5, 5.1.15, 5.3.2, and 5.4.10 of **[IMP-REQS]**). In XMPP this is done by managing one's privacy lists using the 'jabber:iq:privacy' namespace.

Server-side privacy lists enable successful completion of the following use cases:

- Retrieving one's privacy lists.
- Adding, removing, and editing one's privacy lists.
- Setting, changing, or declining active lists.
- Setting, changing, or declining the default list (i.e., the list that is active by default).
- Allowing or blocking messages based on JID, group, or subscription type (or globally).
- Allowing or blocking inbound presence notifications based on JID, group, or subscription type (or globally).
- Allowing or blocking outbound presence notifications based on JID, group, or subscription type (or globally).
- Allowing or blocking IQ stanzas based on JID, group, or subscription type (or globally).
- Allowing or blocking all communications based on JID, group, or subscription type (or globally).

Note: Presence notifications do not include presence subscriptions, only presence information that is broadcasted to entities that are subscribed to a user's presence information. Thus this includes presence stanzas with no 'type' attribute or of type='unavailable' only.

### 10.1. Syntax and Semantics

TOC

A user MAY define one or more privacy lists, which are stored by the user's server. Each <list/> element contains one or more rules in the form of <item/> elements, and each <item/> element uses attributes to define a privacy rule type, a specific value to which the rule applies, the relevant action, and the place of the item in the processing order.

The syntax is as follows:

```
<iq>
  <query xmlns='jabber:iq:privacy'>
    <list name='foo'>
      <item
```

```
          type='[jid|group|subscription]'
          value='bar'
          action='[allow|deny]'
          order='unsignedInt'>
        [<message/>]
        [<presence-in/>]
        [<presence-out/>]
        [<iq/>]
      </item>
    </list>
  </query>
</iq>
```

If the type is "jid", then the 'value' attribute MUST contain a valid Jabber ID. JIDs SHOULD be matched in the following order:

1. <user@domain/resource> (only that resource matches)
2. <user@domain> (any resource matches)
3. <domain/resource> (only that resource matches)
4. <domain> (the domain itself matches, as does any user@domain, domain/resource, or address containing a subdomain)

If the type is "group", then the 'value' attribute SHOULD contain the name of a group in the user's roster. (If a client attempts to update, create, or delete a list item with a group that is not in the user's roster, the server SHOULD return to the client an <item-not-found/> stanza error.)

If the type is "subscription", then the 'value' attribute MUST be one of "both", "to", "from", or "none" as defined under **Roster Syntax and Semantics**, where "none" includes entities that are totally unknown to the user and therefore not in the user's roster at all.

If no 'type' attribute is included, the rule provides the "fall-through" case.

The 'action' attribute MUST be included and its value MUST be either "allow" or "deny".

The 'order' attribute MUST be included and its value MUST be a non-negative integer that is unique among all items in the list. (If a client attempts to create or update a list with non-unique order values, the server MUST return to the client a <bad-request/> stanza error.)

The <item/> element MAY contain one or more child elements that enable an entity to specify more granular control over which kinds of stanzas are to be blocked (i.e., rather than blocking all stanzas). The allowable child elements are:

- <message/> -- blocks incoming message stanzas
- <iq/> -- blocks incoming IQ stanzas
- <presence-in/> -- blocks incoming presence notifications
- <presence-out/> -- blocks outgoing presence notifications

Within the 'jabber:iq:privacy' namespace, the <query/> child of an IQ stanza of type "set" MUST NOT include more than one child element (i.e., the stanza MUST contain only one <active/> element, one <default/> element, or one <list/> element); if a sending entity violates this rule, the receiving entity MUST return a <bad-request/> stanza error.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 554
Keysight, EX-1045
Page 554

When a client adds or updates a privacy list, the <list/> element SHOULD contain at least one <item/> child element; when a client removes a privacy list, the <list/> element MUST NOT contain any <item/> child elements.

When a client updates a privacy list, it must include all of the desired items (i.e., not a "delta").

## 10.2. Business Rules

TOC

1. If there is an active list set for a session, it affects only the session(s) for which it is activated, and only for the duration of the session(s); the server MUST apply the active list only and MUST NOT apply the default list (i.e., there is no "layering" of lists).
2. The default list applies to the user as a whole, and is processed if there is no active list set for the target session/resource to which a stanza is addressed, or if there are no current sessions for the user.
3. If there is no active list set for a session (or there are no current sessions for the user), and there is no default list, then all stanzas SHOULD BE accepted or appropriately processed by the server on behalf of the user in accordance with the **Server Rules for Handling XML Stanzas**.
4. Privacy lists MUST be the first delivery rule applied by a server, superseding (1) the routing and delivery rules specified in **Server Rules for Handling XML Stanzas**, and (2) the handling of subscription-related presence stanzas (and corresponding generation of roster pushes) specified in **Integration of Roster Items and Presence Subscriptions**.
5. The order in which privacy list items are processed by the server is important. List items MUST be processed in ascending order determined by the integer values of the 'order' attribute for each <item/>.
6. As soon as a stanza is matched against a privacy list rule, the server MUST appropriately handle the stanza in accordance with the rule and cease processing.
7. If no fall-through item is provided in a list, the fall-through action is assumed to be "allow".
8. If a user updates the definition for an active list, subsequent processing based on that active list MUST use the updated definition (for all resources to which that active list currently applies).
9. If a change to the subscription state or roster group of a roster item defined in an active or default list occurs during a user's session, subsequent processing based on that list MUST take into account the changed state or group (for all resources to which that list currently applies).
10. When the definition for a rule is modified, the server MUST send an IQ stanza of type "set" to all connected resources, containing a <query/> element with only one <list/> child element, where the 'name' attribute is set to the name of the modified privacy list. These "privacy list pushes" adhere to the same semantics as the "roster pushes" used in roster management, except that only the list name itself (not the full list definition or the "delta") is pushed to the connected resources. It is up to the receiving resource to determine whether to retrieve the modified list definition, although a connected resource SHOULD do so if the list currently applies to it.
11. When a resource attempts to remove a list or specify a new default list while



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 555
Keysight, EX-1045
Page 555

that list applies to a connected resource other than the sending resource, the server MUST return a <conflict/> error to the sending resource and MUST NOT make the requested change.

---

## 10.3. Retrieving One's Privacy Lists

TOC

Example: Client requests names of privacy lists from server:

```
<iq from='romeo@example.net/orchard' type='get' id='getlist1'>
  <query xmlns='jabber:iq:privacy'/>
</iq>
```

Example: Server sends names of privacy lists to client, preceded by active list and default list:

```
<iq type='result' id='getlist1' to='romeo@example.net/orchard'>
  <query xmlns='jabber:iq:privacy'>
    <active name='private'/>
    <default name='public'/>
    <list name='public'/>
    <list name='private'/>
    <list name='special'/>
  </query>
</iq>
```

Example: Client requests a privacy list from server:

```
<iq from='romeo@example.net/orchard' type='get' id='getlist2'>
  <query xmlns='jabber:iq:privacy'>
    <list name='public'/>
  </query>
</iq>
```

Example: Server sends a privacy list to client:

```
<iq type='result' id='getlist2' to='romeo@example.net/orchard'>
  <query xmlns='jabber:iq:privacy'>
    <list name='public'>
      <item type='jid'
            value='tybalt@example.com'
            action='deny'
            order='1'/>
      <item action='allow' order='2'/>
    </list>
  </query>
</iq>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 556
Keysight, EX-1045
Page 556

Example: Client requests another privacy list from server:

```
<iq from='romeo@example.net/orchard' type='get' id='getlist3'>
  <query xmlns='jabber:iq:privacy'>
    <list name='private'/>
  </query>
</iq>
```

Example: Server sends another privacy list to client:

```
<iq type='result' id='getlist3' to='romeo@example.net/orchard'>
  <query xmlns='jabber:iq:privacy'>
    <list name='private'>
      <item type='subscription'
            value='both'
            action='allow'
            order='10'/>
      <item action='deny' order='15'/>
    </list>
  </query>
</iq>
```

Example: Client requests yet another privacy list from server:

```
<iq from='romeo@example.net/orchard' type='get' id='getlist4'>
  <query xmlns='jabber:iq:privacy'>
    <list name='special'/>
  </query>
</iq>
```

Example: Server sends yet another privacy list to client:

```
<iq type='result' id='getlist4' to='romeo@example.net/orchard'>
  <query xmlns='jabber:iq:privacy'>
    <list name='special'>
      <item type='jid'
            value='juliet@example.com'
            action='allow'
            order='6'/>
      <item type='jid'
            value='benvolio@example.org'
            action='allow'
            order='7'/>
      <item type='jid'
            value='mercutio@example.org'
            action='allow'
            order='42'/>
      <item action='deny' order='666'/>
    </list>
  </query>
</iq>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 557
Keysight, EX-1045
Page 557

In this example, the user has three lists: (1) 'public', which allows communications from everyone except one specific entity (this is the default list); (2) 'private', which allows communications only with contacts who have a bidirectional subscription with the user (this is the active list); and (3) 'special', which allows communications only with three specific entities.

If the user attempts to retrieve a list but a list by that name does not exist, the server MUST return an <item-not-found/> stanza error to the user:

Example: Client attempts to retrieve non-existent list:

```
<iq to='romeo@example.net/orchard' type='error' id='getlist5'>
  <query xmlns='jabber:iq:privacy'>
    <list name='The Empty Set'/>
  </query>
  <error type='cancel'>
    <item-not-found
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

The user is allowed to retrieve only one list at a time. If the user attempts to retrieve more than one list in the same request, the server MUST return a <bad request/> stanza error to the user:

Example: Client attempts to retrieve more than one list:

```
<iq to='romeo@example.net/orchard' type='error' id='getlist6'>
  <query xmlns='jabber:iq:privacy'>
    <list name='public'/>
    <list name='private'/>
    <list name='special'/>
  </query>
  <error type='modify'>
    <bad-request
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

## 10.4. Managing Active Lists

TOC

In order to set or change the active list currently being applied by the server, the user MUST send an IQ stanza of type "set" with a <query/> element qualified by the 'jabber:iq:privacy' namespace that contains an empty <active/> child element possessing a 'name' attribute whose value is set to the desired list name.

Example: Client requests change of active list:

```
<iq from='romeo@example.net/orchard' type='set' id='active1'>
  <query xmlns='jabber:iq:privacy'>
```

```
    <active name='special'/>
  </query>
</iq>
```

The server MUST activate and apply the requested list before sending the result back to the client.

Example: Server acknowledges success of active list change:

```
<iq type='result' id='active1' to='romeo@example.net/orchard'/>
```

If the user attempts to set an active list but a list by that name does not exist, the server MUST return an <item-not-found/> stanza error to the user:

Example: Client attempts to set a non-existent list as active:

```
<iq to='romeo@example.net/orchard' type='error' id='active2'>
  <query xmlns='jabber:iq:privacy'>
    <active name='The Empty Set'/>
  </query>
  <error type='cancel'>
    <item-not-found
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

In order to decline the use of any active list, the connected resource MUST send an empty <active/> element with no 'name' attribute.

Example: Client declines the use of active lists:

```
<iq from='romeo@example.net/orchard' type='set' id='active3'>
  <query xmlns='jabber:iq:privacy'>
    <active/>
  </query>
</iq>
```

Example: Server acknowledges success of declining any active list:

```
<iq type='result' id='active3' to='romeo@example.net/orchard'/>
```

## 10.5. Managing the Default List

TOC

In order to change its default list (which applies to the user as a whole, not only the sending resource), the user MUST send an IQ stanza of type "set" with a <query/> element qualified by the 'jabber:iq:privacy' namespace that contains an empty <default/> child element possessing a 'name' attribute whose value is set to the desired



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 559
Keysight, EX-1045
Page 559

list name.

Example: User requests change of default list:

```
<iq from='romeo@example.net/orchard' type='set' id='default1'>
  <query xmlns='jabber:iq:privacy'>
    <default name='special'/>
  </query>
</iq>
```

Example: Server acknowledges success of default list change:

```
<iq type='result' id='default1' to='romeo@example.net/orchard'/>
```

If the user attempts to change which list is the default list but the default list is in use by at least one connected resource other than the sending resource, the server MUST return a <conflict/> stanza error to the sending resource:

Example: Client attempts to change the default list but that list is in use by another resource:

```
<iq to='romeo@example.net/orchard' type='error' id='default1'>
  <query xmlns='jabber:iq:privacy'>
    <default name='special'/>
  </query>
  <error type='cancel'>
    <conflict
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

If the user attempts to set a default list but a list by that name does not exist, the server MUST return an <item-not-found/> stanza error to the user:

Example: Client attempts to set a non-existent list as default:

```
<iq to='romeo@example.net/orchard' type='error' id='default1'>
  <query xmlns='jabber:iq:privacy'>
    <default name='The Empty Set'/>
  </query>
  <error type='cancel'>
    <item-not-found
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

In order to decline the use of a default list (i.e., to use the domain's stanza routing rules at all times), the user MUST send an empty <default/> element with no 'name' attribute.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 560
Keysight, EX-1045
Page 560

Example: Client declines the use of the default list:

```
<iq from='romeo@example.net/orchard' type='set' id='default2'>
  <query xmlns='jabber:iq:privacy'>
    <default/>
  </query>
</iq>
```

Example: Server acknowledges success of declining any default list:

```
<iq type='result' id='default2' to='romeo@example.net/orchard'/>
```

If one connected resource attempts to decline the use of a default list for the user as a whole but the default list currently applies to at least one other connected resource, the server MUST return a <conflict/> error to the sending resource:

Example: Client attempts to decline a default list but that list is in use by another resource:

```
<iq to='romeo@example.net/orchard' type='error' id='default3'>
  <query xmlns='jabber:iq:privacy'>
    <default/>
  </query>
  <error type='cancel'>
    <conflict
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

## 10.6. Editing a Privacy List

TOC

In order to edit a privacy list, the user MUST send an IQ stanza of type "set" with a <query/> element qualified by the 'jabber:iq:privacy' namespace that contains one <list/> child element possessing a 'name' attribute whose value is set to the list name the user would like to edit. The <list/> element MUST contain one or more <item/> elements, which specify the user's desired changes to the list by including all elements in the list (not the "delta").

Example: Client edits a privacy list:

```
<iq from='romeo@example.net/orchard' type='set' id='edit1'>
  <query xmlns='jabber:iq:privacy'>
    <list name='public'>
      <item type='jid'
            value='tybalt@example.com'
            action='deny'
            order='3'/>
      <item type='jid'
            value='paris@example.org'
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 561
Keysight, EX-1045
Page 561

```
            action='deny'
            order='5'/>
      <item action='allow' order='68'/>
    </list>
  </query>
</iq>
```

Example: Server acknowledges success of list edit:

```
<iq type='result' id='edit1' to='romeo@example.net/orchard'/>
```

Note: The value of the 'order' attribute for any given item is not fixed. Thus in the foregoing example if the user would like to add 4 items between the "tybalt@example.com" item and the "paris@example.org" item, the user's client MUST renumber the relevant items before submitting the list to the server.

The server MUST now send a "privacy list push" to all connected resources:

Example: Privacy list push on list edit:

```
<iq to='romeo@example.net/orchard' type='set' id='push1'>
  <query xmlns='jabber:iq:privacy'>
    <list name='public'/>
  </query>
</iq>

<iq to='romeo@example.net/home' type='set' id='push2'>
  <query xmlns='jabber:iq:privacy'>
    <list name='public'/>
  </query>
</iq>
```

In accordance with the semantics of IQ stanzas defined in **[XMPP-CORE]**, each connected resource MUST return an IQ result to the server as well:

Example: Acknowledging receipt of privacy list pushes:

```
<iq from='romeo@example.net/orchard'
    type='result'
    id='push1'/>

<iq from='romeo@example.net/home'
    type='result'
    id='push2'/>
```

## 10.7. Adding a New Privacy List

TOC

The same protocol used to edit an existing list is used to create a new list. If the list name matches that of an existing list, the request to add a new list will overwrite the old



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 562
Keysight, EX-1045
Page 562

one. As with list edits, the server MUST also send a "privacy list push" to all connected resources.

## 10.8. Removing a Privacy List

TOC

In order to remove a privacy list, the user MUST send an IQ stanza of type "set" with a <query/> element qualified by the 'jabber:iq:privacy' namespace that contains one empty <list/> child element possessing a 'name' attribute whose value is set to the list name the user would like to remove.

Example: Client removes a privacy list:

```
<iq from='romeo@example.net/orchard' type='set' id='remove1'>
  <query xmlns='jabber:iq:privacy'>
    <list name='private'/>
  </query>
</iq>
```

Example: Server acknowledges success of list removal:

```
<iq type='result' id='remove1' to='romeo@example.net/orchard'/>
```

If a user attempts to remove a list that is currently being applied to at least one resource other than the sending resource, the server MUST return a <conflict/> stanza error to the user; i.e., the user MUST first set another list to active or default before attempting to remove it. If the user attempts to remove a list but a list by that name does not exist, the server MUST return an <item-not-found/> stanza error to the user. If the user attempts to remove more than one list in the same request, the server MUST return a <bad request/> stanza error to the user.

## 10.9. Blocking Messages

TOC

Server-side privacy lists enable a user to block incoming messages from other entities based on the entity's JID, roster group, or subscription status (or globally). The following examples illustrate the protocol. (Note: For the sake of brevity, IQ stanzas of type "result" are not shown in the following examples, nor are "privacy list pushes".)

Example: User blocks based on JID:

```
<iq from='romeo@example.net/orchard' type='set' id='msg1'>
  <query xmlns='jabber:iq:privacy'>
    <list name='message-jid-example'>
      <item type='jid'
            value='tybalt@example.com'
            action='deny'
            order='3'>
        <message/>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 563
Keysight, EX-1045
Page 563

```
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive messages from the entity with the specified JID.

Example: User blocks based on roster group:

```
<iq from='romeo@example.net/orchard' type='set' id='msg2'>
  <query xmlns='jabber:iq:privacy'>
    <list name='message-group-example'>
      <item type='group'
            value='Enemies'
            action='deny'
            order='4'>
        <message/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive messages from any entities in the specified roster group.

Example: User blocks based on subscription type:

```
<iq from='romeo@example.net/orchard' type='set' id='msg3'>
  <query xmlns='jabber:iq:privacy'>
    <list name='message-sub-example'>
      <item type='subscription'
            value='none'
            action='deny'
            order='5'>
        <message/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive messages from any entities with the specified subscription type.

Example: User blocks globally:

```
<iq from='romeo@example.net/orchard' type='set' id='msg4'>
  <query xmlns='jabber:iq:privacy'>
    <list name='message-global-example'>
      <item action='deny' order='6'>
        <message/>
      </item>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 564
Keysight, EX-1045
Page 564

```
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive messages from any other users.

## 10.10. Blocking Inbound Presence Notifications

TOC

Server-side privacy lists enable a user to block incoming presence notifications from other entities based on the entity's JID, roster group, or subscription status (or globally). The following examples illustrate the protocol.

Note: Presence notifications do not include presence subscriptions, only presence information that is broadcasted to the user because the user is currently subscribed to a contact's presence information. Thus this includes presence stanzas with no 'type' attribute or of type='unavailable' only.

Example: User blocks based on JID:

```
<iq from='romeo@example.net/orchard' type='set' id='presin1'>
  <query xmlns='jabber:iq:privacy'>
    <list name='presin-jid-example'>
      <item type='jid'
            value='tybalt@example.com'
            action='deny'
            order='7'>
        <presence-in/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive presence notifications from the entity with the specified JID.

Example: User blocks based on roster group:

```
<iq from='romeo@example.net/orchard' type='set' id='presin2'>
  <query xmlns='jabber:iq:privacy'>
    <list name='presin-group-example'>
      <item type='group'
            value='Enemies'
            action='deny'
            order='8'>
        <presence-in/>
      </item>
    </list>
  </query>
</iq>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 565
Keysight, EX-1045
Page 565

As a result of creating and applying the foregoing list, the user will not receive presence notifications from any entities in the specified roster group.

Example: User blocks based on subscription type:

```
<iq from='romeo@example.net/orchard' type='set' id='presin3'>
  <query xmlns='jabber:iq:privacy'>
    <list name='presin-sub-example'>
      <item type='subscription'
            value='to'
            action='deny'
            order='9'>
        <presence-in/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive presence notifications from any entities with the specified subscription type.

Example: User blocks globally:

```
<iq from='romeo@example.net/orchard' type='set' id='presin4'>
  <query xmlns='jabber:iq:privacy'>
    <list name='presin-global-example'>
      <item action='deny' order='11'>
        <presence-in/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive presence notifications from any other users.

## 10.11. Blocking Outbound Presence Notifications

TOC

Server-side privacy lists enable a user to block outgoing presence notifications to other entities based on the entity's JID, roster group, or subscription status (or globally). The following examples illustrate the protocol.

Note: Presence notifications do not include presence subscriptions, only presence information that is broadcasted to contacts because those contacts are currently subscribed to the user's presence information. Thus this includes presence stanzas with no 'type' attribute or of type='unavailable' only.

Example: User blocks based on JID:

```
<iq from='romeo@example.net/orchard' type='set' id='presout1'>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 566
Keysight, EX-1045
Page 566

```
  <query xmlns='jabber:iq:privacy'>
    <list name='presout-jid-example'>
      <item type='jid'
            value='tybalt@example.com'
            action='deny'
            order='13'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not send presence notifications to the entity with the specified JID.

Example: User blocks based on roster group:

```
<iq from='romeo@example.net/orchard' type='set' id='presout2'>
  <query xmlns='jabber:iq:privacy'>
    <list name='presout-group-example'>
      <item type='group'
            value='Enemies'
            action='deny'
            order='15'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not send presence notifications to any entities in the specified roster group.

Example: User blocks based on subscription type:

```
<iq from='romeo@example.net/orchard' type='set' id='presout3'>
  <query xmlns='jabber:iq:privacy'>
    <list name='presout-sub-example'>
      <item type='subscription'
            value='from'
            action='deny'
            order='17'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not send presence notifications to any entities with the specified subscription type.

Example: User blocks globally:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 567
Keysight, EX-1045
Page 567

```
<iq from='romeo@example.net/orchard' type='set' id='presout4'>
  <query xmlns='jabber:iq:privacy'>
    <list name='presout-global-example'>
      <item action='deny' order='23'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not send presence notifications to any other users.

## 10.12. Blocking IQ Stanzas

TOC

Server-side privacy lists enable a user to block incoming IQ stanzas from other entities based on the entity's JID, roster group, or subscription status (or globally). The following examples illustrate the protocol.

Example: User blocks based on JID:

```
<iq from='romeo@example.net/orchard' type='set' id='iq1'>
  <query xmlns='jabber:iq:privacy'>
    <list name='iq-jid-example'>
      <item type='jid'
            value='tybalt@example.com'
            action='deny'
            order='29'>
        <iq/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive IQ stanzas from the entity with the specified JID.

Example: User blocks based on roster group:

```
<iq from='romeo@example.net/orchard' type='set' id='iq2'>
  <query xmlns='jabber:iq:privacy'>
    <list name='iq-group-example'>
      <item type='group'
            value='Enemies'
            action='deny'
            order='31'>
        <iq/>
      </item>
    </list>
  </query>
</iq>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 568
Keysight, EX-1045
Page 568

As a result of creating and applying the foregoing list, the user will not receive IQ stanzas from any entities in the specified roster group.

Example: User blocks based on subscription type:

```
<iq from='romeo@example.net/orchard' type='set' id='iq3'>
  <query xmlns='jabber:iq:privacy'>
    <list name='iq-sub-example'>
      <item type='subscription'
            value='none'
            action='deny'
            order='17'>
        <iq/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive IQ stanzas from any entities with the specified subscription type.

Example: User blocks globally:

```
<iq from='romeo@example.net/orchard' type='set' id='iq4'>
  <query xmlns='jabber:iq:privacy'>
    <list name='iq-global-example'>
      <item action='deny' order='1'>
        <iq/>
      </item>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive IQ stanzas from any other users.

## 10.13. Blocking All Communication

TOC

Server-side privacy lists enable a user to block all stanzas from and to other entities based on the entity's JID, roster group, or subscription status (or globally). Note that this includes subscription-related presence stanzas, which are excluded by **Blocking Inbound Presence Notifications**. The following examples illustrate the protocol.

Example: User blocks based on JID:

```
<iq from='romeo@example.net/orchard' type='set' id='all1'>
  <query xmlns='jabber:iq:privacy'>
    <list name='all-jid-example'>
      <item type='jid'
            value='tybalt@example.com'
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 569
Keysight, EX-1045
Page 569

```
            action='deny'
            order='23'/>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive any communications from, nor send any stanzas to, the entity with the specified JID.

Example: User blocks based on roster group:

```
<iq from='romeo@example.net/orchard' type='set' id='all2'>
  <query xmlns='jabber:iq:privacy'>
    <list name='all-group-example'>
      <item type='group'
            value='Enemies'
            action='deny'
            order='13'/>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive any communications from, nor send any stanzas to, any entities in the specified roster group.

Example: User blocks based on subscription type:

```
<iq from='romeo@example.net/orchard' type='set' id='all3'>
  <query xmlns='jabber:iq:privacy'>
    <list name='all-sub-example'>
      <item type='subscription'
            value='none'
            action='deny'
            order='11'/>
    </list>
  </query>
</iq>
```

As a result of creating and applying the foregoing list, the user will not receive any communications from, nor send any stanzas to, any entities with the specified subscription type.

Example: User blocks globally:

```
<iq from='romeo@example.net/orchard' type='set' id='all4'>
  <query xmlns='jabber:iq:privacy'>
    <list name='all-global-example'>
      <item action='deny' order='7'/>
    </list>
  </query>
</iq>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 570
Keysight, EX-1045
Page 570

As a result of creating and applying the foregoing list, the user will not receive any communications from, nor send any stanzas to, any other users.

## 10.14. Blocked Entity Attempts to Communicate with User

TOC

If a blocked entity attempts to send message or presence stanzas to the user, the user's server SHOULD silently drop the stanza and MUST NOT return an error to the sending entity.

If a blocked entity attempts to send an IQ stanza of type "get" or "set" to the user, the user's server MUST return to the sending entity a <service-unavailable/> stanza error, since this is the standard error code sent from a client that does not understand the namespace of an IQ get or set. IQ stanzas of other types SHOULD be silently dropped by the server.

Example: Blocked entity attempts to send IQ get:

```
<iq type='get'
    to='romeo@example.net'
    from='tybalt@example.com/pda'
    id='probing1'>
  <query xmlns='jabber:iq:version'/>
</iq>
```

Example: Server returns error to blocked entity:

```
<iq type='error'
    from='romeo@example.net'
    to='tybalt@example.com/pda'
    id='probing1'>
  <query xmlns='jabber:iq:version'/>
  <error type='cancel'>
    <service-unavailable
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

## 10.15. Higher-Level Heuristics

TOC

When building a representation of a higher-level privacy heuristic, a client SHOULD use the simplest possible representation.

For example, the heuristic "block all communications with any user not in my roster" could be constructed in any of the following ways:

- allow communications from all JIDs in my roster (i.e., listing each JID as a separate list item), but block communications with everyone else
- allow communications from any user who is in one of the groups that make up my roster (i.e., listing each group as a separate list item), but block



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 571
Keysight, EX-1045
Page 571

communications from everyone else
- allow communications from any user with whom I have a subscription of 'both' or 'to' or 'from' (i.e., listing each subscription value separately), but block communications from everyone else
- block communications from anyone whose subscription state is 'none'

The final representation is the simplest and SHOULD be used; here is the XML that would be sent in this case:

```
<iq type='set' id='heuristic1'>
  <query xmlns='jabber:iq:privacy'>
    <list name='heuristic-example'>
      <item type='subscription'
            value='none'
            action='deny'
            order='437'/>
    </list>
  </query>
</iq>
```

## 11. Server Rules for Handling XML Stanzas

TOC

Basic routing and delivery rules for servers are defined in **[XMPP-CORE]**. This section defines additional rules for XMPP-compliant instant messaging and presence servers.

### 11.1. Inbound Stanzas

TOC

If the hostname of the domain identifier portion of the JID contained in the 'to' attribute of an inbound stanza matches a hostname of the server itself and the JID contained in the 'to' attribute is of the form <user@example.com> or <user@example.com/resource>, the server MUST first apply any **privacy lists** that are in force, then follow the rules defined below:

1. If the JID is of the form <user@domain/resource> and an available resource matches the full JID, the recipient's server MUST deliver the stanza to that resource.
2. Else if the JID is of the form <user@domain> or <user@domain/resource> and the associated user account does not exist, the recipient's server (a) SHOULD silently ignore the stanza (i.e., neither deliver it nor return an error) if it is a presence stanza, (b) MUST return a <service-unavailable/> stanza error to the sender if it is an IQ stanza, and (c) SHOULD return a <service-unavailable/> stanza error to the sender if it is a message stanza.
3. Else if the JID is of the form <user@domain/resource> and no available resource matches the full JID, the recipient's server (a) SHOULD silently ignore the stanza (i.e., neither deliver it nor return an error) if it is a presence stanza, (b) MUST return a <service-unavailable/> stanza error to the sender if it is an IQ stanza, and (c) SHOULD treat the stanza as if it were addressed to <user@domain> if it is a message stanza.
4. Else if the JID is of the form <user@domain> and there is at least one



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 572
Keysight, EX-1045
Page 572

available resource available for the user, the recipient's server MUST follow these rules:

1. For message stanzas, the server SHOULD deliver the stanza to the highest-priority available resource (if the resource did not provide a value for the <priority/> element, the server SHOULD consider it to have provided a value of zero). If two or more available resources have the same priority, the server MAY use some other rule (e.g., most recent connect time, most recent activity time, or highest availability as determined by some hierarchy of <show/> values) to choose between them or MAY deliver the message to all such resources. However, the server MUST NOT deliver the stanza to an available resource with a negative priority; if the only available resource has a negative priority, the server SHOULD handle the message as if there were no available resources (defined below). In addition, the server MUST NOT rewrite the 'to' attribute (i.e., it MUST leave it as <user@domain> rather than change it to <user@domain/resource>).
2. For presence stanzas other than those of type "probe", the server MUST deliver the stanza to all available resources; for presence probes, the server SHOULD reply based on the rules defined in **Presence Probes**. In addition, the server MUST NOT rewrite the 'to' attribute (i.e., it MUST leave it as <user@domain> rather than change it to <user@domain/resource>).
3. For IQ stanzas, the server itself MUST reply on behalf of the user with either an IQ result or an IQ error, and MUST NOT deliver the IQ stanza to any of the available resources. Specifically, if the semantics of the qualifying namespace define a reply that the server can provide, the server MUST reply to the stanza on behalf of the user; if not, the server MUST reply with a <service-unavailable/> stanza error.

5. Else if the JID is of the form <user@domain> and there are no available resources associated with the user, how the stanza is handled depends on the stanza type:
    1. For presence stanzas of type "subscribe", "subscribed", "unsubscribe", and "unsubscribed", the server MUST maintain a record of the stanza and deliver the stanza at least once (i.e., when the user next creates an available resource); in addition, the server MUST continue to deliver presence stanzas of type "subscribe" until the user either approves or denies the subscription request (see also **Presence Subscriptions**).
    2. For all other presence stanzas, the server SHOULD silently ignore the stanza by not storing it for later delivery or replying to it on behalf of the user.
    3. For message stanzas, the server MAY choose to store the stanza on behalf of the user and deliver it when the user next becomes available, or forward the message to the user via some other means (e.g., to the user's email



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 573
Keysight, EX-1045
Page 573

account). However, if offline message storage or message forwarding is not enabled, the server MUST return to the sender a <service-unavailable/> stanza error. (Note: Offline message storage and message forwarding are not defined in XMPP, since they are strictly a matter of implementation and service provisioning.)

4. For IQ stanzas, the server itself MUST reply on behalf of the user with either an IQ result or an IQ error. Specifically, if the semantics of the qualifying namespace define a reply that the server can provide, the server MUST reply to the stanza on behalf of the user; if not, the server MUST reply with a <service-unavailable/> stanza error.

## 11.2. Outbound Stanzas

TOC

If the hostname of the domain identifier portion of the address contained in the 'to' attribute of an outbound stanza matches a hostname of the server itself, the server MUST deliver the stanza to a local entity according the rules for **Inbound Stanzas**.

If the hostname of the domain identifier portion of the address contained in the 'to' attribute of an outbound stanza does not match a hostname of the server itself, the server MUST attempt to route the stanza to the foreign domain. The recommended order of actions is as follows:

1. First attempt to resolve the foreign hostname using an **[SRV]** Service of "xmpp-server" and Proto of "tcp", resulting in resource records such as "_xmpp-server._tcp.example.com.", as specified in **[XMPP-CORE]**.
2. If the "xmpp-server" address record resolution fails, attempt to resolve the "_im" or "_pres" **[SRV]** Service as specified in **[IMP-SRV]**, using the "_im" Service for <message/> stanzas and the "_pres" Service for <presence/> stanzas (it is up to the implementation how to handle <iq/> stanzas). This will result in one or more resolutions of the form "_im.<proto>.example.com." or "_pres.<proto>.example.com.", where "<proto>" would be a label registered in the Instant Messaging SRV Protocol Label registry or the Presence SRV Protocol Label registry: either "_xmpp" for an XMPP-aware domain or some other IANA-registered label (e.g., "_simple") for a non-XMPP-aware domain.
3. If both SRV address record resolutions fail, attempt to perform a normal IPv4/IPv6 address record resolution to determine the IP address using the "xmpp-server" port of 5269 registered with the IANA, as specified in **[XMPP-CORE]**.

Administrators of server deployments are strongly encouraged to keep the _im._xmpp, _pres._xmpp, and _xmpp._tcp SRV records properly synchronized, since different implementations might perform the "_im" and "_pres" lookups before the "xmpp-server" lookup.

## 12. IM and Presence Compliance Requirements

TOC



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 574
Keysight, EX-1045
Page 574

This section summarizes the specific aspects of the Extensible Messaging and Presence Protocol that MUST be supported by instant messaging and presence servers and clients in order to be considered compliant implementations. All such applications MUST comply with the requirements specified in **[XMPP-CORE]**. The text in this section specifies additional compliance requirements for instant messaging and presence servers and clients; note well that the requirements described here supplement but do not supersede the core requirements. Note also that a server or client MAY support only presence or instant messaging, and is not required to support both if only a presence service or an instant messaging service is desired.

## 12.1. Servers

TOC

In addition to core server compliance requirements, an instant messaging and presence server MUST additionally support the following protocols:

- All server-related instant messaging and presence syntax and semantics defined in this document, including presence broadcast on behalf of clients, presence subscriptions, roster storage and manipulation, privacy lists, and IM-specific routing and delivery rules

## 12.2. Clients

TOC

In addition to core client compliance requirements, an instant messaging and presence client MUST additionally support the following protocols:

- Generation and handling of the IM-specific semantics of XML stanzas as defined by the XML schemas, including the 'type' attribute of message and presence stanzas as well as their child elements
- All client-related instant messaging syntax and semantics defined in this document, including presence subscriptions, roster management, and privacy lists
- End-to-end object encryption as defined in **End-to-End Object Encryption in the Extensible Messaging and Presence Protocol (XMPP)** [XMPP-E2E]

A client MUST also handle addresses that are encoded as "im:" URIs as specified in **[CPIM]**, and MAY do so by removing the "im:" scheme and entrusting address resolution to the server as specified under **Outbound Stanzas**.

## 13. Internationalization Considerations

TOC

For internationalization considerations, refer to the relevant section of **[XMPP-CORE]**.

## 14. Security Considerations

TOC

Core security considerations for XMPP are defined in the relevant section of



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 575
Keysight, EX-1045
Page 575

**[XMPP-CORE]**.

Additional considerations that apply only to instant messaging and presence applications of XMPP are defined in several places within this memo; specifically:

- When a server processes an inbound stanza of any kind whose intended recipient is a user associated with one of the server's hostnames, the server MUST first apply any **privacy lists** that are in force (see **Server Rules for Handling XML Stanzas**).
- When a server processes an inbound presence stanza of type "probe" whose intended recipient is a user associated with one of the server's hostnames, the server MUST NOT reveal the user's presence information if the sender is an entity that is not authorized to receive that information as determined by presence subscriptions (see **Client and Server Presence Responsibilities**).
- When a server processes an outbound presence stanza with no type or of type "unavailable", it MUST follow the rules defined under **Client and Server Presence Responsibilities** in order to ensure that such presence information is not broadcasted to entities that are not authorized to know such information.
- When a server generates an error stanza in response to receiving a stanza for a user who does not exist, the use of the <service-unavailable/> error condition helps protect against well-known dictionary attacks, since this is the same error condition that is returned if, for instance, the namespace of an IQ child element is not understood, or if offline message storage or message forwarding is not enabled for a domain.

## 15. IANA Considerations

TOC

For a number of related IANA considerations, refer to the relevant section of **[XMPP-CORE]**.

### 15.1. XML Namespace Name for Session Data

TOC

A URN sub-namespace for session-related data in the Extensible Messaging and Presence Protocol (XMPP) is defined as follows. (This namespace name adheres to the format defined in **The IETF XML Registry** [XML-REG].)

URI:
urn:ietf:params:xml:ns:xmpp-session
Specification:
RFC 3921
Description:
This is the XML namespace name for session-related data in the Extensible Messaging and Presence Protocol (XMPP) as defined by RFC 3921.
Registrant Contact:
IETF, XMPP Working Group, <xmppwg@jabber.org>



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 576
Keysight, EX-1045
Page 576

### 15.2. Instant Messaging SRV Protocol Label Registration

TOC

**Address Resolution for Instant Messaging and Presence** [IMP-SRV] defines an Instant Messaging SRV Protocol Label registry for protocols that can provide services that conform to the "_im" SRV Service label. Because XMPP is one such protocol, the IANA registers the "_xmpp" protocol label in the appropriate registry, as follows:

Protocol label:
_xmpp
Specification:
RFC 3921
Description:
Instant messaging protocol label for the Extensible Messaging and Presence Protocol (XMPP) as defined by RFC 3921.
Registrant Contact:
IETF, XMPP Working Group, <xmppwg@jabber.org>

---

### 15.3. Presence SRV Protocol Label Registration

TOC

**Address Resolution for Instant Messaging and Presence** [IMP-SRV] defines a Presence SRV Protocol Label registry for protocols that can provide services that conform to the "_pres" SRV Service label. Because XMPP is one such protocol, the IANA registers the "_xmpp" protocol label in the appropriate registry, as follows:

Protocol label:
_xmpp
Specification:
RFC 3921
Description:
Presence protocol label for the Extensible Messaging and Presence Protocol (XMPP) as defined by RFC 3921.
Registrant Contact:
IETF, XMPP Working Group, <xmppwg@jabber.org>

---

## 16. References

TOC

---

### 16.1. Normative References

TOC


**[CPIM]** Peterson, J., "**Common Profile for Instant Messaging (CPIM)**," RFC 3860, August 2004.

**[IMP-REQS]** **Day, M.**, **Aggarwal, S.**, and **J. Vincent**, "**Instant Messaging / Presence Protocol Requirements**," RFC 2779, February 2000.

**[IMP-SRV]** Peterson, J., "**Address Resolution for Instant Messaging and Presence**," RFC 3861, August 2004.

**[SRV]** **Gulbrandsen, A.**, Vixie, P., and **L. Esibov**, "**A DNS RR for specifying the location of services (DNS SRV)**," RFC 2782, February 2000.

**[TERMS]** **Bradner, S.**, "**Key words for use in RFCs to Indicate Requirement Levels**," BCP 14, RFC 2119, March 1997 (**TXT**, **HTML**, **XML**).

**[XML]** **Bray, T.**, **Paoli, J.**, **Sperberg-McQueen, C.**, and **E. Maler**, "**Extensible Markup Language (XML) 1.0 (2nd ed)**," W3C REC-xml, October 2000.




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 577
Keysight, EX-1045
Page 577

| | |
|---|---|
| **[XML-NAMES]** | **Bray, T.**, **Hollander, D.**, and **A. Layman**, "**Namespaces in XML**," W3C REC-xml-names, January 1999. |
| **[XMPP-CORE]** | Saint-Andre, P., "**Extensible Messaging and Presence Protocol (XMPP): Core**," RFC 3920, October 2004. |
| **[XMPP-E2E]** | Saint-Andre, P., "**End-to-End Signing and Object Encryption for the Extensible Messaging and Presence Protocol (XMPP)**," RFC 3923, October 2004. |


## 16.2. Informative References

TOC


| | |
|---|---|
| **[IMP-MODEL]** | **Day, M.**, **Rosenberg, J.**, and **H. Sugano**, "**A Model for Presence and Instant Messaging**," RFC 2778, February 2000. |
| **[IRC]** | **Oikarinen, J.** and **D. Reed**, "**Internet Relay Chat Protocol**," RFC 1459, May 1993. |
| **[JEP-0045]** | **Saint-Andre, P.**, "**Multi-User Chat**," JSF JEP 0045, June 2004. |
| **[JEP-0054]** | **Saint-Andre, P.**, "**vcard-temp**," JSF JEP 0054, March 2003. |
| **[JEP-0077]** | **Saint-Andre, P.**, "**In-Band Registration**," JSF JEP 0077, August 2004. |
| **[JEP-0078]** | **Saint-Andre, P.**, "**Non-SASL Authentication**," JSF JEP 0078, July 2004. |
| **[JSF]** | **Jabber Software Foundation**, "**Jabber Software Foundation**." |
| **[VCARD]** | **Dawson, F.** and **T. Howes**, "**vCard MIME Directory Profile**," RFC 2426, September 1998 (**HTML**, **XML**). |
| **[XML-REG]** | Mealling, M., "**The IETF XML Registry**," BCP 81, RFC 3688, January 2004. |


## Appendix A. vCards

TOC

Sections 3.1.3 and 4.1.4 of **[IMP-REQS]** require that it be possible to retrieve out-of-band contact information for other users (e.g., telephone number or email address). An XML representation of the vCard specification defined in **RFC 2426** [VCARD] is in common use within the Jabber community to provide such information but is out of scope for XMPP (documentation of this protocol is contained in **[JEP-0054]**, published by the **Jabber Software Foundation** [JSF]).

## Appendix B. XML Schemas

TOC

The following XML schemas are descriptive, not normative. For schemas defining the core features of XMPP, refer to **[XMPP-CORE]**.

### B.1. jabber:client

TOC

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='jabber:client'
    xmlns='jabber:client'
    elementFormDefault='qualified'>

  <xs:import namespace='urn:ietf:params:xml:ns:xmpp-stanzas'/>

  <xs:element name='message'>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 578
Keysight, EX-1045
Page 578

```
    <xs:complexType>
      <xs:sequence>
        <xs:choice minOccurs='0' maxOccurs='unbounded'>
          <xs:element ref='subject'/>
          <xs:element ref='body'/>
          <xs:element ref='thread'/>
        </xs:choice>
        <xs:any     namespace='##other'
                    minOccurs='0'
                    maxOccurs='unbounded'/>
        <xs:element ref='error'
                    minOccurs='0'/>
      </xs:sequence>
      <xs:attribute name='from'
                    type='xs:string'
                    use='optional'/>
      <xs:attribute name='id'
                    type='xs:NMTOKEN'
                    use='optional'/>
      <xs:attribute name='to'
                    type='xs:string'
                    use='optional'/>
      <xs:attribute name='type' use='optional' default='normal'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='chat'/>
            <xs:enumeration value='error'/>
            <xs:enumeration value='groupchat'/>
            <xs:enumeration value='headline'/>
            <xs:enumeration value='normal'/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute ref='xml:lang' use='optional'/>
   </xs:complexType>
</xs:element>

<xs:element name='body'>
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base='xs:string'>
        <xs:attribute ref='xml:lang' use='optional'/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

<xs:element name='subject'>
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base='xs:string'>
        <xs:attribute ref='xml:lang' use='optional'/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

```
<xs:element name='thread' type='xs:NMTOKEN'/>

<xs:element name='presence'>
  <xs:complexType>
    <xs:sequence>
      <xs:choice minOccurs='0' maxOccurs='unbounded'>
        <xs:element ref='show'/>
        <xs:element ref='status'/>
        <xs:element ref='priority'/>
      </xs:choice>
      <xs:any     namespace='##other'
                  minOccurs='0'
                  maxOccurs='unbounded'/>
      <xs:element ref='error'
                  minOccurs='0'/>
    </xs:sequence>
    <xs:attribute name='from'
                  type='xs:string'
                  use='optional'/>
    <xs:attribute name='id'
                  type='xs:NMTOKEN'
                  use='optional'/>
    <xs:attribute name='to'
                  type='xs:string'
                  use='optional'/>
    <xs:attribute name='type' use='optional'>
      <xs:simpleType>
        <xs:restriction base='xs:NCName'>
          <xs:enumeration value='error'/>
          <xs:enumeration value='probe'/>
          <xs:enumeration value='subscribe'/>
          <xs:enumeration value='subscribed'/>
          <xs:enumeration value='unavailable'/>
          <xs:enumeration value='unsubscribe'/>
          <xs:enumeration value='unsubscribed'/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute ref='xml:lang' use='optional'/>
  </xs:complexType>
</xs:element>

<xs:element name='show'>
  <xs:simpleType>
    <xs:restriction base='xs:NCName'>
      <xs:enumeration value='away'/>
      <xs:enumeration value='chat'/>
      <xs:enumeration value='dnd'/>
      <xs:enumeration value='xa'/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>

<xs:element name='status'>
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base='xs:string'>
```

```
        <xs:attribute ref='xml:lang' use='optional'/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

<xs:element name='priority' type='xs:byte'/>

<xs:element name='iq'>
  <xs:complexType>
    <xs:sequence>
      <xs:any     namespace='##other'
                  minOccurs='0'/>
      <xs:element ref='error'
                  minOccurs='0'/>
    </xs:sequence>
    <xs:attribute name='from'
                  type='xs:string'
                  use='optional'/>
    <xs:attribute name='id'
                  type='xs:NMTOKEN'
                  use='required'/>
    <xs:attribute name='to'
                  type='xs:string'
                  use='optional'/>
    <xs:attribute name='type' use='required'>
      <xs:simpleType>
        <xs:restriction base='xs:NCName'>
          <xs:enumeration value='error'/>
          <xs:enumeration value='get'/>
          <xs:enumeration value='result'/>
          <xs:enumeration value='set'/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute ref='xml:lang' use='optional'/>
  </xs:complexType>
</xs:element>

<xs:element name='error'>
  <xs:complexType>
    <xs:sequence  xmlns:err='urn:ietf:params:xml:ns:xmpp-stanzas'>
      <xs:group   ref='err:stanzaErrorGroup'/>
      <xs:element ref='err:text'
                  minOccurs='0'/>
    </xs:sequence>
    <xs:attribute name='code' type='xs:byte' use='optional'/>
    <xs:attribute name='type' use='required'>
      <xs:simpleType>
        <xs:restriction base='xs:NCName'>
          <xs:enumeration value='auth'/>
          <xs:enumeration value='cancel'/>
          <xs:enumeration value='continue'/>
          <xs:enumeration value='modify'/>
          <xs:enumeration value='wait'/>
        </xs:restriction>
      </xs:simpleType>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 581
**Keysight, EX-1045**
**Page 581**

```
      </xs:attribute>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

## B.2. jabber:server

TOC

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='jabber:server'
    xmlns='jabber:server'
    elementFormDefault='qualified'>

  <xs:import namespace='urn:ietf:params:xml:ns:xmpp-stanzas'/>

  <xs:element name='message'>
     <xs:complexType>
        <xs:sequence>
          <xs:choice minOccurs='0' maxOccurs='unbounded'>
            <xs:element ref='subject'/>
            <xs:element ref='body'/>
            <xs:element ref='thread'/>
          </xs:choice>
          <xs:any     namespace='##other'
                      minOccurs='0'
                      maxOccurs='unbounded'/>
          <xs:element ref='error'
                      minOccurs='0'/>
        </xs:sequence>
        <xs:attribute name='from'
                      type='xs:string'
                      use='required'/>
        <xs:attribute name='id'
                      type='xs:NMTOKEN'
                      use='optional'/>
        <xs:attribute name='to'
                      type='xs:string'
                      use='required'/>
        <xs:attribute name='type' use='optional' default='normal'>
          <xs:simpleType>
            <xs:restriction base='xs:NCName'>
              <xs:enumeration value='chat'/>
              <xs:enumeration value='error'/>
              <xs:enumeration value='groupchat'/>
              <xs:enumeration value='headline'/>
              <xs:enumeration value='normal'/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute ref='xml:lang' use='optional'/>
     </xs:complexType>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 582
Keysight, EX-1045
Page 582

```
  </xs:element>

  <xs:element name='body'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:string'>
          <xs:attribute ref='xml:lang' use='optional'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name='subject'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:string'>
          <xs:attribute ref='xml:lang' use='optional'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name='thread' type='xs:NMTOKEN'/>

  <xs:element name='presence'>
    <xs:complexType>
      <xs:sequence>
        <xs:choice minOccurs='0' maxOccurs='unbounded'>
          <xs:element ref='show'/>
          <xs:element ref='status'/>
          <xs:element ref='priority'/>
        </xs:choice>
        <xs:any     namespace='##other'
                    minOccurs='0'
                    maxOccurs='unbounded'/>
        <xs:element ref='error'
                    minOccurs='0'/>
      </xs:sequence>
      <xs:attribute name='from'
                    type='xs:string'
                    use='required'/>
      <xs:attribute name='id'
                    type='xs:NMTOKEN'
                    use='optional'/>
      <xs:attribute name='to'
                    type='xs:string'
                    use='required'/>
      <xs:attribute name='type' use='optional'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='error'/>
            <xs:enumeration value='probe'/>
            <xs:enumeration value='subscribe'/>
            <xs:enumeration value='subscribed'/>
            <xs:enumeration value='unavailable'/>
            <xs:enumeration value='unsubscribe'/>
            <xs:enumeration value='unsubscribed'/>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 583
Keysight, EX-1045
Page 583

```
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute ref='xml:lang' use='optional'/>
  </xs:complexType>
</xs:element>

<xs:element name='show'>
  <xs:simpleType>
    <xs:restriction base='xs:NCName'>
      <xs:enumeration value='away'/>
      <xs:enumeration value='chat'/>
      <xs:enumeration value='dnd'/>
      <xs:enumeration value='xa'/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>

<xs:element name='status'>
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base='xs:string'>
        <xs:attribute ref='xml:lang' use='optional'/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

<xs:element name='priority' type='xs:byte'/>

<xs:element name='iq'>
  <xs:complexType>
    <xs:sequence>
      <xs:any     namespace='##other'
                  minOccurs='0'/>
      <xs:element ref='error'
                  minOccurs='0'/>
    </xs:sequence>
    <xs:attribute name='from'
                  type='xs:string'
                  use='required'/>
    <xs:attribute name='id'
                  type='xs:NMTOKEN'
                  use='required'/>
    <xs:attribute name='to'
                  type='xs:string'
                  use='required'/>
    <xs:attribute name='type' use='required'>
      <xs:simpleType>
        <xs:restriction base='xs:NCName'>
          <xs:enumeration value='error'/>
          <xs:enumeration value='get'/>
          <xs:enumeration value='result'/>
          <xs:enumeration value='set'/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
```



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 584
Keysight, EX-1045
Page 584

```
      <xs:attribute ref='xml:lang' use='optional'/>
    </xs:complexType>
  </xs:element>

  <xs:element name='error'>
    <xs:complexType>
      <xs:sequence  xmlns:err='urn:ietf:params:xml:ns:xmpp-stanzas'>
        <xs:group   ref='err:stanzaErrorGroup'/>
        <xs:element ref='err:text'
                    minOccurs='0'/>
      </xs:sequence>
      <xs:attribute name='code' type='xs:byte' use='optional'/>
      <xs:attribute name='type' use='required'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='auth'/>
            <xs:enumeration value='cancel'/>
            <xs:enumeration value='continue'/>
            <xs:enumeration value='modify'/>
            <xs:enumeration value='wait'/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

## B.3. session

TOC

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:ietf:params:xml:ns:xmpp-session'
    xmlns='urn:ietf:params:xml:ns:xmpp-session'
    elementFormDefault='qualified'>

  <xs:element name='session' type='empty'/>

  <xs:simpleType name='empty'>
    <xs:restriction base='xs:string'>
      <xs:enumeration value=''/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

## B.4. jabber:iq:privacy

TOC

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='jabber:iq:privacy'
    xmlns='jabber:iq:privacy'
    elementFormDefault='qualified'>

  <xs:element name='query'>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref='active'
                    minOccurs='0'/>
        <xs:element ref='default'
                    minOccurs='0'/>
        <xs:element ref='list'
                    minOccurs='0'
                    maxOccurs='unbounded'/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name='active'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:NMTOKEN'>
          <xs:attribute name='name'
                        type='xs:string'
                        use='optional'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name='default'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:NMTOKEN'>
          <xs:attribute name='name'
                        type='xs:string'
                        use='optional'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name='list'>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref='item'
                    minOccurs='0'
                    maxOccurs='unbounded'/>
      </xs:sequence>
      <xs:attribute name='name'
                    type='xs:string'
                    use='required'/>
```

```
    </xs:complexType>
  </xs:element>

  <xs:element name='item'>
    <xs:complexType>
      <xs:sequence>
        <xs:element name='iq'
                    minOccurs='0'
                    type='empty'/>
        <xs:element name='message'
                    minOccurs='0'
                    type='empty'/>
        <xs:element name='presence-in'
                    minOccurs='0'
                    type='empty'/>
        <xs:element name='presence-out'
                    minOccurs='0'
                    type='empty'/>
      </xs:sequence>
      <xs:attribute name='action' use='required'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='allow'/>
            <xs:enumeration value='deny'/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name='order'
                    type='xs:unsignedInt'
                    use='required'/>
      <xs:attribute name='type' use='optional'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='group'/>
            <xs:enumeration value='jid'/>
            <xs:enumeration value='subscription'/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name='value'
                    type='xs:string'
                    use='optional'/>
    </xs:complexType>
  </xs:element>

  <xs:simpleType name='empty'>
    <xs:restriction base='xs:string'>
      <xs:enumeration value=''/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

## B.5. jabber:iq:roster

TOC



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 587
Keysight, EX-1045
Page 587

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='jabber:iq:roster'
    xmlns='jabber:iq:roster'
    elementFormDefault='qualified'>

  <xs:element name='query'>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref='item'
                    minOccurs='0'
                    maxOccurs='unbounded'/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name='item'>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref='group'
                    minOccurs='0'
                    maxOccurs='unbounded'/>
      </xs:sequence>
      <xs:attribute name='ask' use='optional'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='subscribe'/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name='jid' type='xs:string' use='required'/>
      <xs:attribute name='name' type='xs:string' use='optional'/>
      <xs:attribute name='subscription' use='optional'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='both'/>
            <xs:enumeration value='from'/>
            <xs:enumeration value='none'/>
            <xs:enumeration value='remove'/>
            <xs:enumeration value='to'/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>

  <xs:element name='group' type='xs:string'/>

</xs:schema>
```

## Appendix C. Differences Between Jabber IM/Presence Protocols and XMPP

TOC



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 588
Keysight, EX-1045
Page 588

This section is non-normative.

XMPP has been adapted from the protocols originally developed in the Jabber open-source community, which can be thought of as "XMPP 0.9". Because there exists a large installed base of Jabber implementations and deployments, it may be helpful to specify the key differences between the relevant Jabber protocols and XMPP in order to expedite and encourage upgrades of those implementations and deployments to XMPP. This section summarizes the differences that relate specifically to instant messaging and presence applications, while the corresponding section of **[XMPP-CORE]** summarizes the differences that relate to all XMPP applications.

## C.1. Session Establishment

TOC

The client-to-server authentication protocol developed in the Jabber community assumed that every client is an IM client and therefore initiated an IM session upon successful authentication and resource binding, which are performed simultaneously (documention of this protocol is contained in **[JEP-0078]**, published by the **Jabber Software Foundation** [JSF]). XMPP maintains a stricter separation between core functionality and IM functionality; therefore, an IM session is not created until the client specifically requests one using the protocol defined under **Session Establishment**.

## C.2. Privacy Lists

TOC

The Jabber community began to define a protocol for communications blocking (privacy lists) in late 2001, but that effort was deprecated once the XMPP Working Group was formed. Therefore the protocol defined under **Blocking Communication** is the only such protocol defined for use in the Jabber community.

## Appendix D. Contributors

TOC

Most of the core aspects of the Extensible Messaging and Presence Protocol were developed originally within the Jabber open-source community in 1999. This community was founded by Jeremie Miller, who released source code for the initial version of the jabberd server in January 1999. Major early contributors to the base protocol also included Ryan Eatmon, Peter Millard, Thomas Muldowney, and Dave Smith. Work specific to instant messaging and presence by the XMPP Working Group has concentrated especially on IM session establishment and communication blocking (privacy lists); the session establishment protocol was mainly developed by Rob Norris and Joe Hildebrand, and the privacy lists protocol was originally contributed by Peter Millard.

## Appendix E. Acknowledgements

TOC

Thanks are due to a number of individuals in addition to the contributors listed. Although it is difficult to provide a complete list, the following individuals were particularly helpful



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 589
Keysight, EX-1045
Page 589

in defining the protocols or in commenting on the specifications in this memo: Thomas Charron, Richard Dobson, Schuyler Heath, Jonathan Hogg, Craig Kaes, Jacek Konieczny, Lisa Dusseault, Alexey Melnikov, Keith Minkler, Julian Missig, Pete Resnick, Marshall Rose, Jean-Louis Seguineau, Alexey Shchepin, Iain Shigeoka, and David Waite. Thanks also to members of the XMPP Working Group and the IETF community for comments and feedback provided throughout the life of this memo.

## Author's Address

Peter Saint-Andre (editor)
Jabber Software Foundation
**Email:** stpeter@jabber.org

## Full Copyright Statement

Copyright © The Internet Society (2004).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at **http://www.ietf.org/ipr**.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at **ietf-ipr@ietf.org**.

## Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).

darknet.org.uk
Don't Learn to HACK - Hack to LEARN

SCAN YOUR WEBSITE FOR XSS VULNERABILITIES WITH ACUNETIX WVS FREE EDITION DOWNLOAD

HOME | ABOUT DARKNET | POPULAR POSTS | DARKNET ARCHIVES | CONTACT DARKNET

*27 August 2012 | 650 views*

# XMPPloit – A Tool to Attack XMPP Connections




XMPPloit is a command-line tool to attack XMPP connections, allowing the attacker to place a gateway between the client and the server and perform different attacks on the client stream.

The tool exploit implements vulnerabilities at the client & server side utilizing the XMPP protocol.

The main goal is that all the process is transparently for the user and never replace any certificate (like HTTPS attacks).

**Features**

- Downgrade the authentication mechanism (can obtain the user credentials)
- Force the client not to use an encrypted communication
- Set filters for traffic manipulation

Filters that have been implemented in this version for Google Talk are:

- Read all the the user's account mails
- Read and modify all the user's account contacts (being or not in the roster).

You can download XMPPloit here:

XMPPloit.7z

Or read more here.

Tags: blackhat. blackhat 2012, luis delgado, xmpp, xmpp exploit, xmpp hacking, xmpp security,

SEARCH DARKNET

Search for: Enter Keyword

SUBSCRIBE

36384 readers BY FEEDBURNER

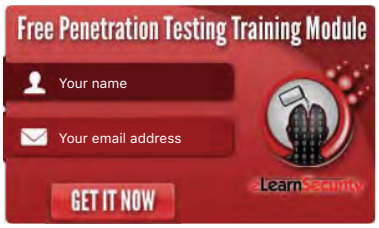
Enter your Email

Subscribe me!

404 Not Found

nginx/1.15.8

FREEBIES!




NAVIGATION

- About Darknet
- Popular Posts
- Darknet Tags
- Darknet Archives
- Contact Darknet
- Advertise
- Hack Tools/Exploits

xmpp vulnerability, xmpploit

Posted in: Hacking Tools, Network Hacking | Add a Comment

**Subscribe to Darknet RSS Feed**

**Recent in Hacking Tools:**
- chapcrack – A tool for parsing and decrypting MS-CHAPv2 network handshakes.
- Hcon Security Testing Framework (HconSTF) v0.4 – Fire Base
- The Mole v0.3 Released For Download – Automatic SQL Injection Exploitation Tool

**Related Posts:**
- Slowloris – HTTP DoS Tool in PERL
- SSLyze v0.4 Released – Scan & Analyze SSL Server Configuration
- Apple Fixes 'Misleading' Leopard Firewall Settings

**Most Read in Hacking Tools:**
- Top 15 Security/Hacking Tools & Utilities - 1,727,429 views
- Brutus Password Cracker – Download brutus-aet2.zip AET2 - 959,157 views
- wwwhack 1.9 – Download wwwhack19.zip Web Hacking Tool - 553,308 views

Advertise Here NOW!
Click for details..

## Leave a Reply

NAME (REQUIRED)

MAIL (WILL NOT BE PUBLISHED) (REQUIRED)

WEBSITE

SUBMIT COMMENT

NOTIFY ME OF FOLLOWUP COMMENTS VIA E-MAIL

**SPONSORS**






Advertise on Darknet

**USERS ONLINE**

6 Users Online [ Stats ]

**RECENT COMMENTS**

- Shasta Mozzone on Sophos Offers Free Android Antivirus App
- NNM on Hcon Security Testing Framework (HconSTF) v0.4 – Fire Base
- IT Security on Hcon Security Testing Framework (HconSTF) v0.4 – Fire Base
- David Williams on Nvidia Investigates Claims Of Online Store Compromise During Spate Of Hacking
- Prashant Pandey on MySQL 1 Liner Hack Gives Root Access Without Password
- Johan on Windows XML Core Services Exploit Attacked In The Wild – CVE-2012-1889

**RECENT ARTICLES**

- XMPPloit – A Tool to Attack XMPP Connections
- Microsoft Patches Critical Security Vulnerabilities In Windows, Office, IE, Exchange & SQL Server
- chapcrack – A tool for parsing and decrypting MS-CHAPv2 network handshakes.

- Sophos Offers Free Android Antivirus App
- Hcon Security Testing Framework (HconSTF) v0.4 – Fire Base
- Nvidia Investigates Claims Of Online Store Compromise During Spate Of Hacking
- spt v0.6.0 – Simple Phishing Toolkit Available For Download
- Yahoo! Voices Hacked With SQL Injection – Passwords In Plaintext
- Microsoft Enhanced Mitigation Evaluation Toolkit (EMET) 3rd Party GUI
- Android Malware App Covertly Makes Purchases On China Mobile Market

**TOPICS**

- Apple
- Authors
- Countermeasures
- Cryptography
- Database Hacking
- Events/Cons
- Exploits/Vulnerabilities
- Forensics
- General Hacking
- General News
- Hacking Tools
- Hardware Hacking
- Legal Issues
- Linux Hacking
- Malware
- Network Hacking
- Old Skool Philes
- Password Cracking
- Phishing
- Privacy
- Programming
- Retards
- Security Software
- Site News
- Social Engineering
- Spammers & Scammers
- Telecomms Hacking



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 593
**Keysight, EX-1045**
**Page 593**

- UNIX Hacking
- Virology
- Web Hacking
- Windows Hacking
- Wireless Hacking

**TAGS**

botnets computer-security darknet Database Hacking dos exploits fuzzing google hacking hacking-networks hacking-web-applications hacking-websites hacking-windows hacking tool Hacking Tools Information-Security information gathering Legal Issues malware microsoft network-security Network Hacking Password Cracking penetration-testing Phishing Privacy scammers Security Security Software spam spammers sql-injection trojan trojans virus viruses vulnerabilities web-application-security web-security Web Hacking windows windows-security Windows Hacking worms XSS

**FACEBOOK**

Sign Up Create an account or log in to see what your friends are doing.

Facebook social plugin



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 594
**Keysight, EX-1045**
**Page 594**

**TWITTER**

1 Million Accounts Leaked From Banks, Government Agencies & Consultancy Firms: Seems like some hactivists have b... http://t.co/nQW3kaYU 2385 days ago

XMPPloit – A Tool to Attack XMPP Connections: XMPPloit is a command-line tool to attack XMPP connections, allowi... http://t.co/78hjEleP 2387 days ago

Microsoft Patches Critical Security Vulnerabilities In Windows, Office, IE, Exchange & SQL Server: Another huge ... http://t.co/SQry6y3W 2397 days ago

@sptoolkit thanks! 2398 days ago

Follow us on Twitter

© Darknet – The Darkside 2000-2012. Don't Learn to HACK - Hack to LEARN.

Network Working Group
Request for Comments: 3551
Obsoletes: 1890
Category: Standards Track

H. Schulzrinne
Columbia University
S. Casner
Packet Design

July 2003

# RTP Profile for Audio and Video Conferences with Minimal Control

## Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Copyright Notice

Copyright (C) The Internet Society (2003). All Rights Reserved.

## Abstract

This document describes a profile called "RTP/AVP" for the use of the real-time transport protocol (RTP), version 2, and the associated control protocol, RTCP, within audio and video multiparticipant conferences with minimal control. It provides interpretations of generic fields within the RTP specification suitable for audio and video conferences. In particular, this document defines a set of default mappings from payload type numbers to encodings.

This document also describes how audio and video data may be carried within RTP. It defines a set of standard encodings and their names when used within RTP. The descriptions provide pointers to reference implementations and the detailed standards. This document is meant as an aid for implementors of audio, video and other real-time multimedia applications.

This memorandum obsoletes RFC 1890. It is mostly backwards-compatible except for functions removed because two interoperable implementations were not found. The additions to RFC 1890 codify existing practice in the use of payload formats under this profile and include new payload formats defined since RFC 1890 was published.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 596
Keysight, EX-1045
Page 596

Table of Contents


1. Introduction ..... 3
   1.1 Terminology ..... 3
2. RTP and RTCP Packet Forms and Protocol Behavior ..... 4
3. Registering Additional Encodings ..... 6
4. Audio ..... 8
   4.1 Encoding-Independent Rules ..... 8
   4.2 Operating Recommendations ..... 9
   4.3 Guidelines for Sample-Based Audio Encodings ..... 10
   4.4 Guidelines for Frame-Based Audio Encodings ..... 11
   4.5 Audio Encodings ..... 12
      4.5.1 DVI4 ..... 13
      4.5.2 G722 ..... 14
      4.5.3 G723 ..... 14
      4.5.4 G726-40, G726-32, G726-24, and G726-16 ..... 18
      4.5.5 G728 ..... 19
      4.5.6 G729 ..... 20
      4.5.7 G729D and G729E ..... 22
      4.5.8 GSM ..... 24
      4.5.9 GSM-EFR ..... 27
      4.5.10 L8 ..... 27
      4.5.11 L16 ..... 27
      4.5.12 LPC ..... 27
      4.5.13 MPA ..... 28
      4.5.14 PCMA and PCMU ..... 28
      4.5.15 QCELP ..... 28
      4.5.16 RED ..... 29
      4.5.17 VDVI ..... 29
5. Video ..... 30
   5.1 CelB ..... 30
   5.2 JPEG ..... 30
   5.3 H261 ..... 30
   5.4 H263 ..... 31
   5.5 H263-1998 ..... 31
   5.6 MPV ..... 31
   5.7 MP2T ..... 31
   5.8 nv ..... 32
6. Payload Type Definitions ..... 32
7. RTP over TCP and Similar Byte Stream Protocols ..... 34
8. Port Assignment ..... 34
9. Changes from RFC 1890 ..... 35
10. Security Considerations ..... 38
11. IANA Considerations ..... 39
12. References ..... 39
   12.1 Normative References ..... 39
   12.2 Informative References ..... 39
13. Current Locations of Related Resources ..... 41




https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 597
Keysight, EX-1045
Page 597

14. Acknowledgments ..... 42
15. Intellectual Property Rights Statement ..... 43
16. Authors' Addresses ..... 43
17. Full Copyright Statement ..... 44

## 1. Introduction

This profile defines aspects of RTP left unspecified in the RTP Version 2 protocol definition (RFC 3550) [1]. This profile is intended for the use within audio and video conferences with minimal session control. In particular, no support for the negotiation of parameters or membership control is provided. The profile is expected to be useful in sessions where no negotiation or membership control are used (e.g., using the static payload types and the membership indications provided by RTCP), but this profile may also be useful in conjunction with a higher-level control protocol.

Use of this profile may be implicit in the use of the appropriate applications; there may be no explicit indication by port number, protocol identifier or the like. Applications such as session directories may use the name for this profile specified in Section 11.

Other profiles may make different choices for the items specified here.

This document also defines a set of encodings and payload formats for audio and video. These payload format descriptions are included here only as a matter of convenience since they are too small to warrant separate documents. Use of these payload formats is NOT REQUIRED to use this profile. Only the binding of some of the payload formats to static payload type numbers in Tables 4 and 5 is normative.

### 1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [2] and indicate requirement levels for implementations compliant with this RTP profile.

This document defines the term media type as dividing encodings of audio and video content into three classes: audio, video and audio/video (interleaved).



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 598
Keysight, EX-1045
Page 598

## 2. RTP and RTCP Packet Forms and Protocol Behavior

The section "RTP Profiles and Payload Format Specifications" of RFC 3550 enumerates a number of items that can be specified or modified in a profile. This section addresses these items. Generally, this profile follows the default and/or recommended aspects of the RTP specification.

RTP data header: The standard format of the fixed RTP data header is used (one marker bit).

Payload types: Static payload types are defined in Section 6.

RTP data header additions: No additional fixed fields are appended to the RTP data header.

RTP data header extensions: No RTP header extensions are defined, but applications operating under this profile MAY use such extensions. Thus, applications SHOULD NOT assume that the RTP header X bit is always zero and SHOULD be prepared to ignore the header extension. If a header extension is defined in the future, that definition MUST specify the contents of the first 16 bits in such a way that multiple different extensions can be identified.

RTCP packet types: No additional RTCP packet types are defined by this profile specification.

RTCP report interval: The suggested constants are to be used for the RTCP report interval calculation. Sessions operating under this profile MAY specify a separate parameter for the RTCP traffic bandwidth rather than using the default fraction of the session bandwidth. The RTCP traffic bandwidth MAY be divided into two separate session parameters for those participants which are active data senders and those which are not. Following the recommendation in the RTP specification [1] that 1/4 of the RTCP bandwidth be dedicated to data senders, the RECOMMENDED default values for these two parameters would be 1.25% and 3.75%, respectively. For a particular session, the RTCP bandwidth for non-data-senders MAY be set to zero when operating on unidirectional links or for sessions that don't require feedback on the quality of reception. The RTCP bandwidth for data senders SHOULD be kept non-zero so that sender reports can still be sent for inter-media synchronization and to identify the source by CNAME. The means by which the one or two session parameters for RTCP bandwidth are specified is beyond the scope of this memo.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 599
Keysight, EX-1045
Page 599

SR/RR extension: No extension section is defined for the RTCP SR or RR packet.

SDES use: Applications MAY use any of the SDES items described in the RTP specification. While CNAME information MUST be sent every reporting interval, other items SHOULD only be sent every third reporting interval, with NAME sent seven out of eight times within that slot and the remaining SDES items cyclically taking up the eighth slot, as defined in Section 6.2.2 of the RTP specification. In other words, NAME is sent in RTCP packets 1, 4, 7, 10, 13, 16, 19, while, say, EMAIL is used in RTCP packet 22.

Security: The RTP default security services are also the default under this profile.

String-to-key mapping: No mapping is specified by this profile.

Congestion: RTP and this profile may be used in the context of enhanced network service, for example, through Integrated Services (RFC 1633) [4] or Differentiated Services (RFC 2475) [5], or they may be used with best effort service.

If enhanced service is being used, RTP receivers SHOULD monitor packet loss to ensure that the service that was requested is actually being delivered. If it is not, then they SHOULD assume that they are receiving best-effort service and behave accordingly.

If best-effort service is being used, RTP receivers SHOULD monitor packet loss to ensure that the packet loss rate is within acceptable parameters. Packet loss is considered acceptable if a TCP flow across the same network path and experiencing the same network conditions would achieve an average throughput, measured on a reasonable timescale, that is not less than the RTP flow is achieving. This condition can be satisfied by implementing congestion control mechanisms to adapt the transmission rate (or the number of layers subscribed for a layered multicast session), or by arranging for a receiver to leave the session if the loss rate is unacceptably high.

The comparison to TCP cannot be specified exactly, but is intended as an "order-of-magnitude" comparison in timescale and throughput. The timescale on which TCP throughput is measured is the round-trip time of the connection. In essence, this requirement states that it is not acceptable to deploy an application (using RTP or any other transport protocol) on the best-effort Internet which consumes bandwidth arbitrarily and does not compete fairly with TCP within an order of magnitude.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 600
Keysight, EX-1045
Page 600

Underlying protocol: The profile specifies the use of RTP over unicast and multicast UDP as well as TCP. (This does not preclude the use of these definitions when RTP is carried by other lower-layer protocols.)

Transport mapping: The standard mapping of RTP and RTCP to transport-level addresses is used.

Encapsulation: This profile leaves to applications the specification of RTP encapsulation in protocols other than UDP.

## 3. Registering Additional Encodings

This profile lists a set of encodings, each of which is comprised of a particular media data compression or representation plus a payload format for encapsulation within RTP. Some of those payload formats are specified here, while others are specified in separate RFCs. It is expected that additional encodings beyond the set listed here will be created in the future and specified in additional payload format RFCs.

This profile also assigns to each encoding a short name which MAY be used by higher-level control protocols, such as the Session Description Protocol (SDP), RFC 2327 [6], to identify encodings selected for a particular RTP session.

In some contexts it may be useful to refer to these encodings in the form of a MIME content-type. To facilitate this, RFC 3555 [7] provides registrations for all of the encodings names listed here as MIME subtype names under the "audio" and "video" MIME types through the MIME registration procedure as specified in RFC 2048 [8].

Any additional encodings specified for use under this profile (or others) may also be assigned names registered as MIME subtypes with the Internet Assigned Numbers Authority (IANA). This registry provides a means to insure that the names assigned to the additional encodings are kept unique. RFC 3555 specifies the information that is required for the registration of RTP encodings.

In addition to assigning names to encodings, this profile also assigns static RTP payload type numbers to some of them. However, the payload type number space is relatively small and cannot accommodate assignments for all existing and future encodings. During the early stages of RTP development, it was necessary to use statically assigned payload types because no other mechanism had been specified to bind encodings to payload types. It was anticipated that non-RTP means beyond the scope of this memo (such as directory services or invitation protocols) would be specified to establish a



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 601
Keysight, EX-1045
Page 601

dynamic mapping between a payload type and an encoding. Now, mechanisms for defining dynamic payload type bindings have been specified in the Session Description Protocol (SDP) and in other protocols such as ITU-T Recommendation H.323/H.245. These mechanisms associate the registered name of the encoding/payload format, along with any additional required parameters, such as the RTP timestamp clock rate and number of channels, with a payload type number. This association is effective only for the duration of the RTP session in which the dynamic payload type binding is made. This association applies only to the RTP session for which it is made, thus the numbers can be re-used for different encodings in different sessions so the number space limitation is avoided.

This profile reserves payload type numbers in the range 96-127 exclusively for dynamic assignment. Applications SHOULD first use values in this range for dynamic payload types. Those applications which need to define more than 32 dynamic payload types MAY bind codes below 96, in which case it is RECOMMENDED that unassigned payload type numbers be used first. However, the statically assigned payload types are default bindings and MAY be dynamically bound to new encodings if needed. Redefining payload types below 96 may cause incorrect operation if an attempt is made to join a session without obtaining session description information that defines the dynamic payload types.

Dynamic payload types SHOULD NOT be used without a well-defined mechanism to indicate the mapping. Systems that expect to interoperate with others operating under this profile SHOULD NOT make their own assignments of proprietary encodings to particular, fixed payload types.

This specification establishes the policy that no additional static payload types will be assigned beyond the ones defined in this document. Establishing this policy avoids the problem of trying to create a set of criteria for accepting static assignments and encourages the implementation and deployment of the dynamic payload type mechanisms.

The final set of static payload type assignments is provided in Tables 4 and 5.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 602
**Keysight, EX-1045**
**Page 602**

## 4. Audio

### 4.1 Encoding-Independent Rules

Since the ability to suppress silence is one of the primary motivations for using packets to transmit voice, the RTP header carries both a sequence number and a timestamp to allow a receiver to distinguish between lost packets and periods of time when no data was transmitted. Discontiguous transmission (silence suppression) MAY be used with any audio payload format. Receivers MUST assume that senders may suppress silence unless this is restricted by signaling specified elsewhere. (Even if the transmitter does not suppress silence, the receiver should be prepared to handle periods when no data is present since packets may be lost.)

Some payload formats (see Sections 4.5.3 and 4.5.6) define a "silence insertion descriptor" or "comfort noise" frame to specify parameters for artificial noise that may be generated during a period of silence to approximate the background noise at the source. For other payload formats, a generic Comfort Noise (CN) payload format is specified in RFC 3389 [9]. When the CN payload format is used with another payload format, different values in the RTP payload type field distinguish comfort-noise packets from those of the selected payload format.

For applications which send either no packets or occasional comfort-noise packets during silence, the first packet of a talkspurt, that is, the first packet after a silence period during which packets have not been transmitted contiguously, SHOULD be distinguished by setting the marker bit in the RTP data header to one. The marker bit in all other packets is zero. The beginning of a talkspurt MAY be used to adjust the playout delay to reflect changing network delays. Applications without silence suppression MUST set the marker bit to zero.

The RTP clock rate used for generating the RTP timestamp is independent of the number of channels and the encoding; it usually equals the number of sampling periods per second. For N-channel encodings, each sampling period (say, 1/8,000 of a second) generates N samples. (This terminology is standard, but somewhat confusing, as the total number of samples generated per second is then the sampling rate times the channel count.)

If multiple audio channels are used, channels are numbered left-to-right, starting at one. In RTP audio packets, information from lower-numbered channels precedes that from higher-numbered channels.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 603
Keysight, EX-1045
Page 603

For more than two channels, the convention followed by the AIFF-C audio interchange format SHOULD be followed [3], using the following notation, unless some other convention is specified for a particular encoding or payload format:

l left
r right
c center
S surround
F front
R rear

| channels | description | channel 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| 2 | stereo | l | r | | | | |
| 3 | | l | r | c | | | |
| 4 | | l | c | r | S | | |
| 5 | | Fl | Fr | Fc | Sl | Sr | |
| 6 | | l | lc | c | r | rc | S |

Note: RFC 1890 defined two conventions for the ordering of four audio channels. Since the ordering is indicated implicitly by the number of channels, this was ambiguous. In this revision, the order described as "quadrophonic" has been eliminated to remove the ambiguity. This choice was based on the observation that quadrophonic consumer audio format did not become popular whereas surround-sound subsequently has.

Samples for all channels belonging to a single sampling instant MUST be within the same packet. The interleaving of samples from different channels depends on the encoding. General guidelines are given in Section 4.3 and 4.4.

The sampling frequency SHOULD be drawn from the set: 8,000, 11,025, 16,000, 22,050, 24,000, 32,000, 44,100 and 48,000 Hz. (Older Apple Macintosh computers had a native sample rate of 22,254.54 Hz, which can be converted to 22,050 with acceptable quality by dropping 4 samples in a 20 ms frame.) However, most audio encodings are defined for a more restricted set of sampling frequencies. Receivers SHOULD be prepared to accept multi-channel audio, but MAY choose to only play a single channel.

### 4.2 Operating Recommendations

The following recommendations are default operating parameters. Applications SHOULD be prepared to handle other values. The ranges given are meant to give guidance to application writers, allowing a



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 604
Keysight, EX-1045
Page 604

set of applications conforming to these guidelines to interoperate without additional negotiation. These guidelines are not intended to restrict operating parameters for applications that can negotiate a set of interoperable parameters, e.g., through a conference control protocol.

For packetized audio, the default packetization interval SHOULD have a duration of 20 ms or one frame, whichever is longer, unless otherwise noted in Table 1 (column "ms/packet"). The packetization interval determines the minimum end-to-end delay; longer packets introduce less header overhead but higher delay and make packet loss more noticeable. For non-interactive applications such as lectures or for links with severe bandwidth constraints, a higher packetization delay MAY be used. A receiver SHOULD accept packets representing between 0 and 200 ms of audio data. (For framed audio encodings, a receiver SHOULD accept packets with a number of frames equal to 200 ms divided by the frame duration, rounded up.) This restriction allows reasonable buffer sizing for the receiver.

### 4.3 Guidelines for Sample-Based Audio Encodings

In sample-based encodings, each audio sample is represented by a fixed number of bits. Within the compressed audio data, codes for individual samples may span octet boundaries. An RTP audio packet may contain any number of audio samples, subject to the constraint that the number of bits per sample times the number of samples per packet yields an integral octet count. Fractional encodings produce less than one octet per sample.

The duration of an audio packet is determined by the number of samples in the packet.

For sample-based encodings producing one or more octets per sample, samples from different channels sampled at the same sampling instant SHOULD be packed in consecutive octets. For example, for a two-channel encoding, the octet sequence is (left channel, first sample), (right channel, first sample), (left channel, second sample), (right channel, second sample), .... For multi-octet encodings, octets SHOULD be transmitted in network byte order (i.e., most significant octet first).

The packing of sample-based encodings producing less than one octet per sample is encoding-specific.

The RTP timestamp reflects the instant at which the first sample in the packet was sampled, that is, the oldest information in the packet.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 605
Keysight, EX-1045
Page 605

### 4.4 Guidelines for Frame-Based Audio Encodings

Frame-based encodings encode a fixed-length block of audio into another block of compressed data, typically also of fixed length. For frame-based encodings, the sender MAY choose to combine several such frames into a single RTP packet. The receiver can tell the number of frames contained in an RTP packet, if all the frames have the same length, by dividing the RTP payload length by the audio frame size which is defined as part of the encoding. This does not work when carrying frames of different sizes unless the frame sizes are relatively prime. If not, the frames MUST indicate their size.

For frame-based codecs, the channel order is defined for the whole block. That is, for two-channel audio, right and left samples SHOULD be coded independently, with the encoded frame for the left channel preceding that for the right channel.

All frame-oriented audio codecs SHOULD be able to encode and decode several consecutive frames within a single packet. Since the frame size for the frame-oriented codecs is given, there is no need to use a separate designation for the same encoding, but with different number of frames per packet.

RTP packets SHALL contain a whole number of frames, with frames inserted according to age within a packet, so that the oldest frame (to be played first) occurs immediately after the RTP packet header. The RTP timestamp reflects the instant at which the first sample in the first frame was sampled, that is, the oldest information in the packet.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 606
Keysight, EX-1045
Page 606

### 4.5 Audio Encodings

| name of encoding | sample/frame | bits/sample | sampling rate | ms/frame | default ms/packet |
|---|---|---|---|---|---|
| DVI4 | sample | 4 | var. | | 20 |
| G722 | sample | 8 | 16,000 | | 20 |
| G723 | frame | N/A | 8,000 | 30 | 30 |
| G726-40 | sample | 5 | 8,000 | | 20 |
| G726-32 | sample | 4 | 8,000 | | 20 |
| G726-24 | sample | 3 | 8,000 | | 20 |
| G726-16 | sample | 2 | 8,000 | | 20 |
| G728 | frame | N/A | 8,000 | 2.5 | 20 |
| G729 | frame | N/A | 8,000 | 10 | 20 |
| G729D | frame | N/A | 8,000 | 10 | 20 |
| G729E | frame | N/A | 8,000 | 10 | 20 |
| GSM | frame | N/A | 8,000 | 20 | 20 |
| GSM-EFR | frame | N/A | 8,000 | 20 | 20 |
| L8 | sample | 8 | var. | | 20 |
| L16 | sample | 16 | var. | | 20 |
| LPC | frame | N/A | 8,000 | 20 | 20 |
| MPA | frame | N/A | var. | var. | |
| PCMA | sample | 8 | var. | | 20 |
| PCMU | sample | 8 | var. | | 20 |
| QCELP | frame | N/A | 8,000 | 20 | 20 |
| VDVI | sample | var. | var. | | 20 |

Table 1: Properties of Audio Encodings (N/A: not applicable; var.: variable)

The characteristics of the audio encodings described in this document are shown in Table 1; they are listed in order of their payload type in Table 4. While most audio codecs are only specified for a fixed sampling rate, some sample-based algorithms (indicated by an entry of "var." in the sampling rate column of Table 1) may be used with different sampling rates, resulting in different coded bit rates. When used with a sampling rate other than that for which a static payload type is defined, non-RTP means beyond the scope of this memo MUST be used to define a dynamic payload type and MUST indicate the selected RTP timestamp clock rate, which is usually the same as the sampling rate for audio.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 607
Keysight, EX-1045
Page 607

### 4.5.1 DVI4

DVI4 uses an adaptive delta pulse code modulation (ADPCM) encoding scheme that was specified by the Interactive Multimedia Association (IMA) as the "IMA ADPCM wave type". However, the encoding defined here as DVI4 differs in three respects from the IMA specification:

- The RTP DVI4 header contains the predicted value rather than the first sample value contained the IMA ADPCM block header.
- IMA ADPCM blocks contain an odd number of samples, since the first sample of a block is contained just in the header (uncompressed), followed by an even number of compressed samples. DVI4 has an even number of compressed samples only, using the `predict' word from the header to decode the first sample.
- For DVI4, the 4-bit samples are packed with the first sample in the four most significant bits and the second sample in the four least significant bits. In the IMA ADPCM codec, the samples are packed in the opposite order.

Each packet contains a single DVI block. This profile only defines the 4-bit-per-sample version, while IMA also specified a 3-bit-per-sample encoding.

The "header" word for each channel has the following structure:

```
int16  predict;  /* predicted value of first sample
                    from the previous block (L16 format) */
u_int8 index;    /* current index into stepsize table */
u_int8 reserved; /* set to zero by sender, ignored by receiver */
```

Each octet following the header contains two 4-bit samples, thus the number of samples per packet MUST be even because there is no means to indicate a partially filled last octet.

Packing of samples for multiple channels is for further study.

The IMA ADPCM algorithm was described in the document IMA Recommended Practices for Enhancing Digital Audio Compatibility in Multimedia Systems (version 3.0). However, the Interactive Multimedia Association ceased operations in 1997. Resources for an archived copy of that document and a software implementation of the RTP DVI4 encoding are listed in Section 13.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 608
Keysight, EX-1045
Page 608

### 4.5.2 G722

G722 is specified in ITU-T Recommendation G.722, "7 kHz audio-coding within 64 kbit/s". The G.722 encoder produces a stream of octets, each of which SHALL be octet-aligned in an RTP packet. The first bit transmitted in the G.722 octet, which is the most significant bit of the higher sub-band sample, SHALL correspond to the most significant bit of the octet in the RTP packet.

Even though the actual sampling rate for G.722 audio is 16,000 Hz, the RTP clock rate for the G722 payload format is 8,000 Hz because that value was erroneously assigned in RFC 1890 and must remain unchanged for backward compatibility. The octet rate or sample-pair rate is 8,000 Hz.

### 4.5.3 G723

G723 is specified in ITU Recommendation G.723.1, "Dual-rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s". The G.723.1 5.3/6.3 kbit/s codec was defined by the ITU-T as a mandatory codec for ITU-T H.324 GSTN videophone terminal applications. The algorithm has a floating point specification in Annex B to G.723.1, a silence compression algorithm in Annex A to G.723.1 and a scalable channel coding scheme for wireless applications in G.723.1 Annex C.

This Recommendation specifies a coded representation that can be used for compressing the speech signal component of multi-media services at a very low bit rate. Audio is encoded in 30 ms frames, with an additional delay of 7.5 ms due to look-ahead. A G.723.1 frame can be one of three sizes: 24 octets (6.3 kb/s frame), 20 octets (5.3 kb/s frame), or 4 octets. These 4-octet frames are called SID frames (Silence Insertion Descriptor) and are used to specify comfort noise parameters. There is no restriction on how 4, 20, and 24 octet frames are intermixed. The least significant two bits of the first octet in the frame determine the frame size and codec type:

| bits | content | octets/frame |
|---|---|---|
| 00 | high-rate speech (6.3 kb/s) | 24 |
| 01 | low-rate speech (5.3 kb/s) | 20 |
| 10 | SID frame | 4 |
| 11 | reserved | |



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 609
Keysight, EX-1045
Page 609

It is possible to switch between the two rates at any 30 ms frame boundary. Both (5.3 kb/s and 6.3 kb/s) rates are a mandatory part of the encoder and decoder. Receivers MUST accept both data rates and MUST accept SID frames unless restriction of these capabilities has been signaled. The MIME registration for G723 in RFC 3555 [7] specifies parameters that MAY be used with MIME or SDP to restrict to a single data rate or to restrict the use of SID frames. This coder was optimized to represent speech with near-toll quality at the above rates using a limited amount of complexity.

The packing of the encoded bit stream into octets and the transmission order of the octets is specified in Rec. G.723.1 and is the same as that produced by the G.723 C code reference implementation. For the 6.3 kb/s data rate, this packing is illustrated as follows, where the header (HDR) bits are always "0 0" as shown in Fig. 1 to indicate operation at 6.3 kb/s, and the Z bit is always set to zero. The diagrams show the bit packing in "network byte order", also known as big-endian order. The bits of each 32-bit word are numbered 0 to 31, with the most significant bit on the left and numbered 0. The octets (bytes) of each word are transmitted most significant octet first. The bits of each data field are numbered in the order of the bit stream representation of the encoding (least significant bit first). The vertical bars indicate the boundaries between field fragments.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 610
Keysight, EX-1045
Page 610

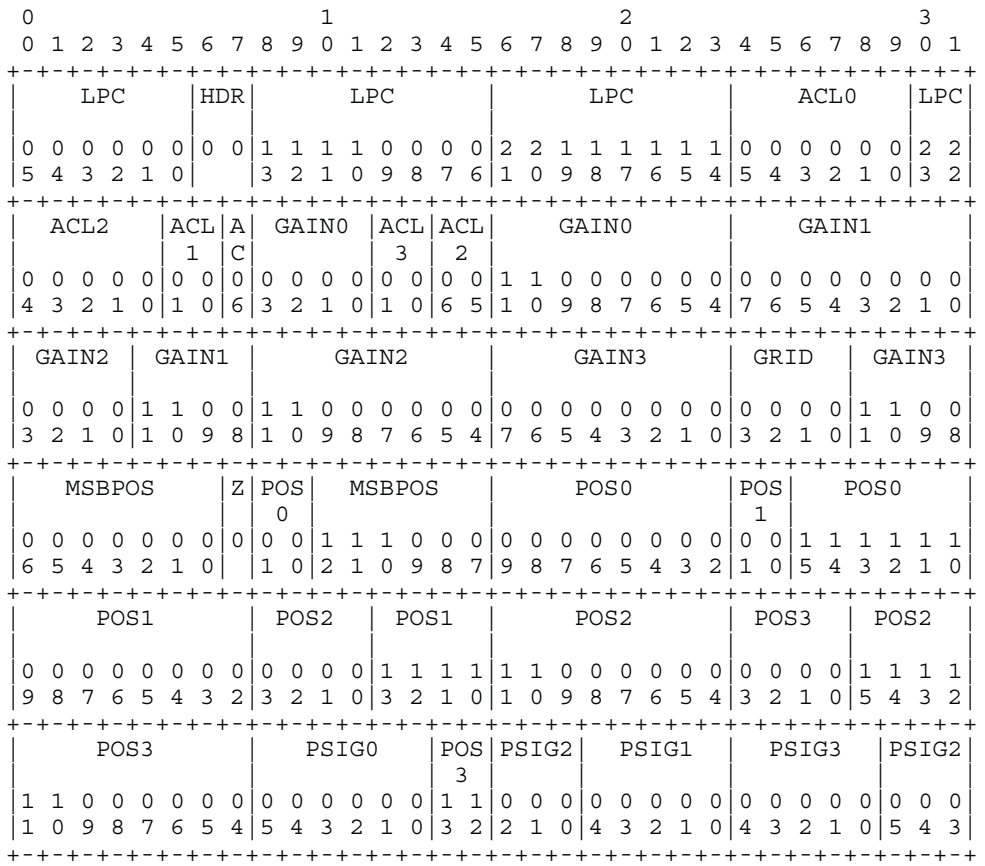



Figure 1: G.723 (6.3 kb/s) bit packing

For the 5.3 kb/s data rate, the header (HDR) bits are always "0 1", as shown in Fig. 2, to indicate operation at 5.3 kb/s.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 611
Keysight, EX-1045
Page 611

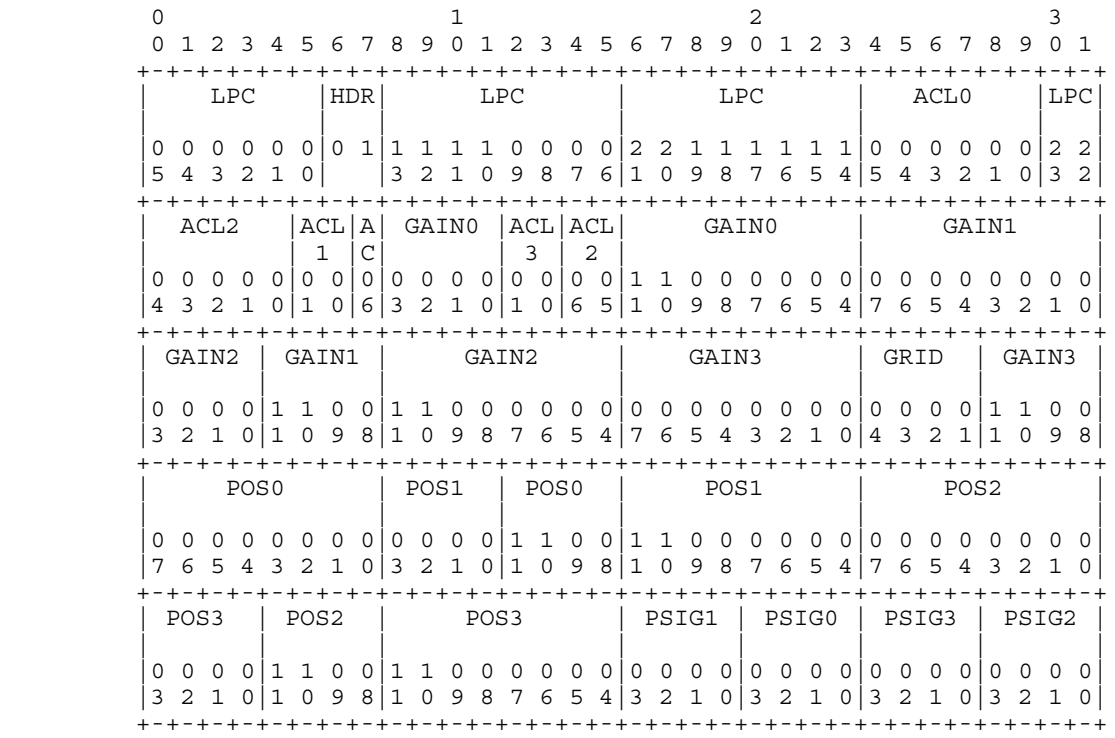



Figure 2: G.723 (5.3 kb/s) bit packing

The packing of G.723.1 SID (silence) frames, which are indicated by the header (HDR) bits having the pattern "1 0", is depicted in Fig. 3.

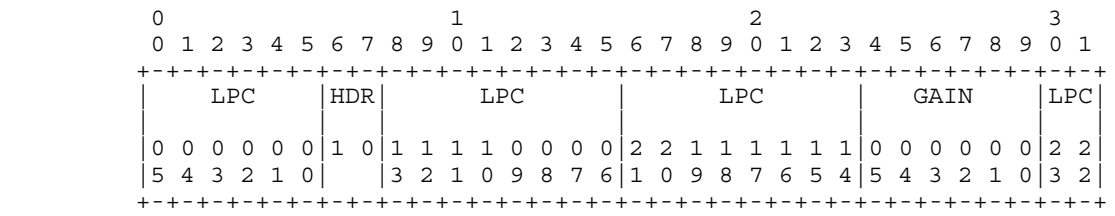



Figure 3: G.723 SID mode bit packing



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 612
Keysight, EX-1045
Page 612

### 4.5.4 G726-40, G726-32, G726-24, and G726-16

ITU-T Recommendation G.726 describes, among others, the algorithm recommended for conversion of a single 64 kbit/s A-law or mu-law PCM channel encoded at 8,000 samples/sec to and from a 40, 32, 24, or 16 kbit/s channel. The conversion is applied to the PCM stream using an Adaptive Differential Pulse Code Modulation (ADPCM) transcoding technique. The ADPCM representation consists of a series of codewords with a one-to-one correspondence to the samples in the PCM stream. The G726 data rates of 40, 32, 24, and 16 kbit/s have codewords of 5, 4, 3, and 2 bits, respectively.

The 16 and 24 kbit/s encodings do not provide toll quality speech. They are designed for used in overloaded Digital Circuit Multiplication Equipment (DCME). ITU-T G.726 recommends that the 16 and 24 kbit/s encodings should be alternated with higher data rate encodings to provide an average sample size of between 3.5 and 3.7 bits per sample.

The encodings of G.726 are here denoted as G726-40, G726-32, G726-24, and G726-16. Prior to 1990, G721 described the 32 kbit/s ADPCM encoding, and G723 described the 40, 32, and 16 kbit/s encodings. Thus, G726-32 designates the same algorithm as G721 in RFC 1890.

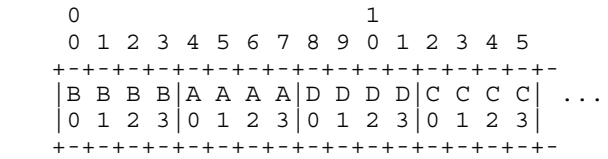
A stream of G726 codewords contains no information on the encoding being used, therefore transitions between G726 encoding types are not permitted within a sequence of packed codewords. Applications MUST determine the encoding type of packed codewords from the RTP payload identifier.

No payload-specific header information SHALL be included as part of the audio data. A stream of G726 codewords MUST be packed into octets as follows: the first codeword is placed into the first octet such that the least significant bit of the codeword aligns with the least significant bit in the octet, the second codeword is then packed so that its least significant bit coincides with the least significant unoccupied bit in the octet. When a complete codeword cannot be placed into an octet, the bits overlapping the octet boundary are placed into the least significant bits of the next octet. Packing MUST end with a completely packed final octet. The number of codewords packed will therefore be a multiple of 8, 2, 8, and 4 for G726-40, G726-32, G726-24, and G726-16, respectively. An example of the packing scheme for G726-32 codewords is as shown, where bit 7 is the least significant bit of the first octet, and bit A3 is the least significant bit of the first codeword:

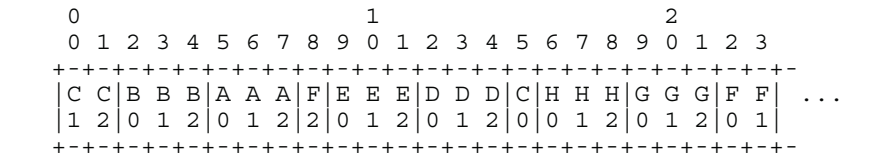


https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 613
Keysight, EX-1045
Page 613




An example of the packing scheme for G726-24 codewords follows, where again bit 7 is the least significant bit of the first octet, and bit A2 is the least significant bit of the first codeword:




Note that the "little-endian" direction in which samples are packed into octets in the G726-16, -24, -32 and -40 payload formats specified here is consistent with ITU-T Recommendation X.420, but is the opposite of what is specified in ITU-T Recommendation I.366.2 Annex E for ATM AAL2 transport. A second set of RTP payload formats matching the packetization of I.366.2 Annex E and identified by MIME subtypes AAL2-G726-16, -24, -32 and -40 will be specified in a separate document.

### 4.5.5 G728

G728 is specified in ITU-T Recommendation G.728, "Coding of speech at 16 kbit/s using low-delay code excited linear prediction".

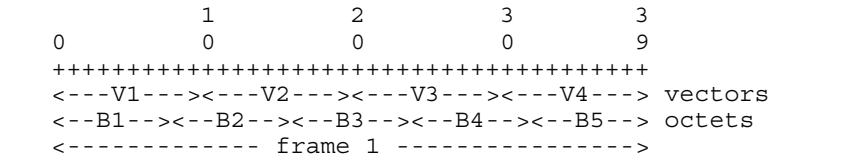
A G.278 encoder translates 5 consecutive audio samples into a 10-bit codebook index, resulting in a bit rate of 16 kb/s for audio sampled at 8,000 samples per second. The group of five consecutive samples is called a vector. Four consecutive vectors, labeled V1 to V4 (where V1 is to be played first by the receiver), build one G.728 frame. The four vectors of 40 bits are packed into 5 octets, labeled B1 through B5. B1 SHALL be placed first in the RTP packet.

Referring to the figure below, the principle for bit order is "maintenance of bit significance". Bits from an older vector are more significant than bits from newer vectors. The MSB of the frame goes to the MSB of B1 and the LSB of the frame goes to LSB of B5.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 614
Keysight, EX-1045
Page 614




In particular, B1 contains the eight most significant bits of V1, with the MSB of V1 being the MSB of B1. B2 contains the two least significant bits of V1, the more significant of the two in its MSB, and the six most significant bits of V2. B1 SHALL be placed first in the RTP packet and B5 last.

### 4.5.6 G729

G729 is specified in ITU-T Recommendation G.729, "Coding of speech at 8 kbit/s using conjugate structure-algebraic code excited linear prediction (CS-ACELP)". A reduced-complexity version of the G.729 algorithm is specified in Annex A to Rec. G.729. The speech coding algorithms in the main body of G.729 and in G.729 Annex A are fully interoperable with each other, so there is no need to further distinguish between them. An implementation that signals or accepts use of G729 payload format may implement either G.729 or G.729A unless restricted by additional signaling specified elsewhere related specifically to the encoding rather than the payload format. The G.729 and G.729 Annex A codecs were optimized to represent speech with high quality, where G.729 Annex A trades some speech quality for an approximate 50% complexity reduction [10]. See the next Section (4.5.7) for other data rates added in later G.729 Annexes. For all data rates, the sampling frequency (and RTP timestamp clock rate) is 8,000 Hz.

A voice activity detector (VAD) and comfort noise generator (CNG) algorithm in Annex B of G.729 is RECOMMENDED for digital simultaneous voice and data applications and can be used in conjunction with G.729 or G.729 Annex A. A G.729 or G.729 Annex A frame contains 10 octets, while the G.729 Annex B comfort noise frame occupies 2 octets. Receivers MUST accept comfort noise frames if restriction of their use has not been signaled. The MIME registration for G729 in RFC 3555 [7] specifies a parameter that MAY be used with MIME or SDP to restrict the use of comfort noise frames.

A G729 RTP packet may consist of zero or more G.729 or G.729 Annex A frames, followed by zero or one G.729 Annex B frames. The presence of a comfort noise frame can be deduced from the length of the RTP payload. The default packetization interval is 20 ms (two frames), but in some situations it may be desirable to send 10 ms packets. An



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 615
Keysight, EX-1045
Page 615

example would be a transition from speech to comfort noise in the first 10 ms of the packet. For some applications, a longer packetization interval may be required to reduce the packet rate.

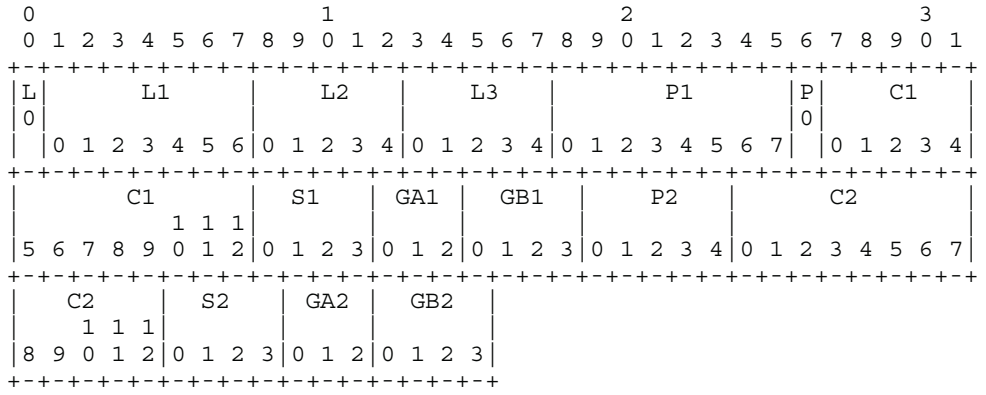



Figure 4: G.729 and G.729A bit packing

The transmitted parameters of a G.729/G.729A 10-ms frame, consisting of 80 bits, are defined in Recommendation G.729, Table 8/G.729. The mapping of the these parameters is given below in Fig. 4. The diagrams show the bit packing in "network byte order", also known as big-endian order. The bits of each 32-bit word are numbered 0 to 31, with the most significant bit on the left and numbered 0. The octets (bytes) of each word are transmitted most significant octet first. The bits of each data field are numbered in the order as produced by the G.729 C code reference implementation.

The packing of the G.729 Annex B comfort noise frame is shown in Fig. 5.

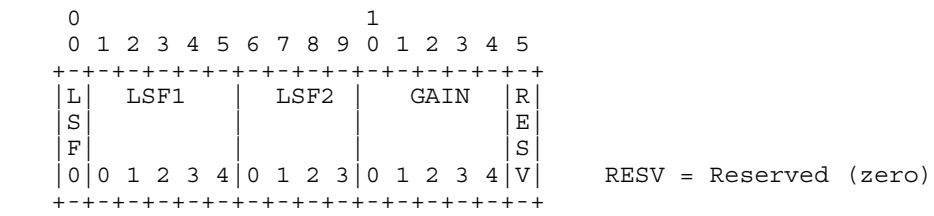



Figure 5: G.729 Annex B bit packing



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 616
Keysight, EX-1045
Page 616

### 4.5.7 G729D and G729E

Annexes D and E to ITU-T Recommendation G.729 provide additional data rates. Because the data rate is not signaled in the bitstream, the different data rates are given distinct RTP encoding names which are mapped to distinct payload type numbers. G729D indicates a 6.4 kbit/s coding mode (G.729 Annex D, for momentary reduction in channel capacity), while G729E indicates an 11.8 kbit/s mode (G.729 Annex E, for improved performance with a wide range of narrow-band input signals, e.g., music and background noise). Annex E has two operating modes, backward adaptive and forward adaptive, which are signaled by the first two bits in each frame (the most significant two bits of the first octet).

The voice activity detector (VAD) and comfort noise generator (CNG) algorithm specified in Annex B of G.729 may be used with Annex D and Annex E frames in addition to G.729 and G.729 Annex A frames. The algorithm details for the operation of Annexes D and E with the Annex B CNG are specified in G.729 Annexes F and G. Note that Annexes F and G do not introduce any new encodings. Receivers MUST accept comfort noise frames if restriction of their use has not been signaled. The MIME registrations for G729D and G729E in RFC 3555 [7] specify a parameter that MAY be used with MIME or SDP to restrict the use of comfort noise frames.

For G729D, an RTP packet may consist of zero or more G.729 Annex D frames, followed by zero or one G.729 Annex B frame. Similarly, for G729E, an RTP packet may consist of zero or more G.729 Annex E frames, followed by zero or one G.729 Annex B frame. The presence of a comfort noise frame can be deduced from the length of the RTP payload.

A single RTP packet must contain frames of only one data rate, optionally followed by one comfort noise frame. The data rate may be changed from packet to packet by changing the payload type number. G.729 Annexes D, E and H describe what the encoding and decoding algorithms must do to accommodate a change in data rate.

For G729D, the bits of a G.729 Annex D frame are formatted as shown below in Fig. 6 (cf. Table D.1/G.729). The frame length is 64 bits.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 617
Keysight, EX-1045
Page 617

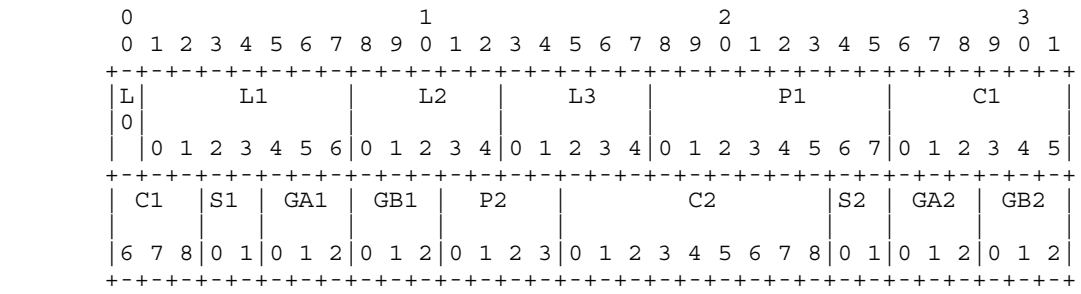



Figure 6: G.729 Annex D bit packing

The net bit rate for the G.729 Annex E algorithm is 11.8 kbit/s and a total of 118 bits are used. Two bits are appended as "don't care" bits to complete an integer number of octets for the frame. For G729E, the bits of a data frame are formatted as shown in the next two diagrams (cf. Table E.1/G.729). The fields for the G729E forward adaptive mode are packed as shown in Fig. 7.

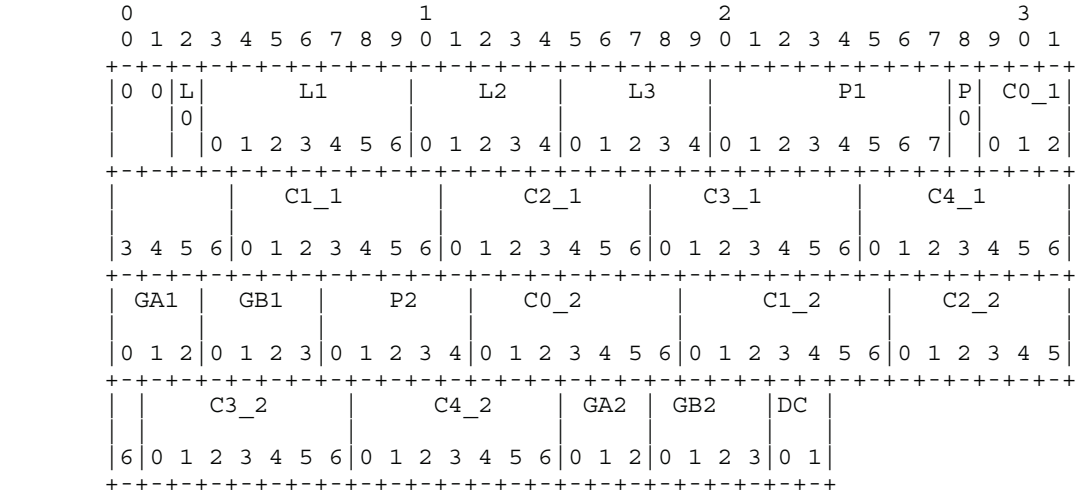



Figure 7: G.729 Annex E (forward adaptive mode) bit packing

The fields for the G729E backward adaptive mode are packed as shown in Fig. 8.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 618
Keysight, EX-1045
Page 618

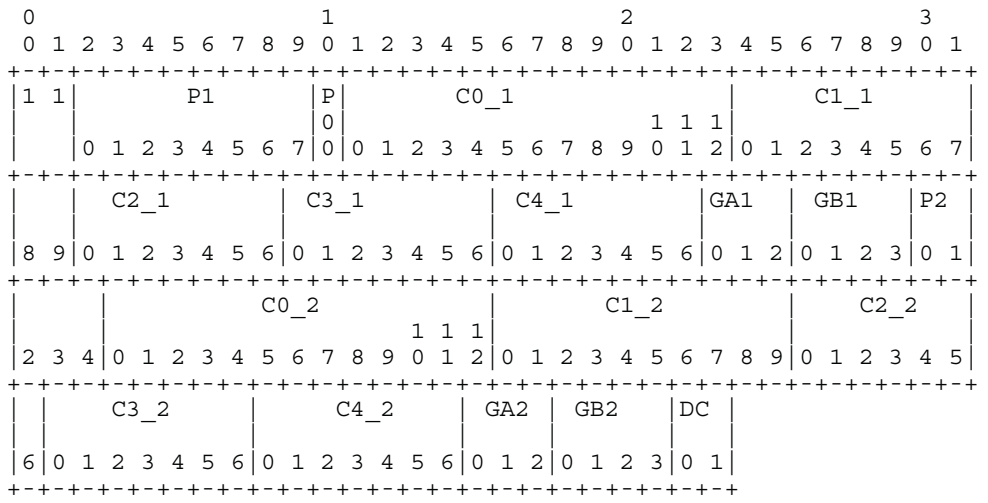



Figure 8: G.729 Annex E (backward adaptive mode) bit packing

### 4.5.8 GSM

GSM (Group Speciale Mobile) denotes the European GSM 06.10 standard for full-rate speech transcoding, ETS 300 961, which is based on RPE/LTP (residual pulse excitation/long term prediction) coding at a rate of 13 kb/s [11,12,13]. The text of the standard can be obtained from:

ETSI (European Telecommunications Standards Institute)
ETSI Secretariat: B.P.152
F-06561 Valbonne Cedex
France
Phone: +33 92 94 42 00
Fax: +33 93 65 47 16

Blocks of 160 audio samples are compressed into 33 octets, for an effective data rate of 13,200 b/s.

#### 4.5.8.1 General Packaging Issues

The GSM standard (ETS 300 961) specifies the bit stream produced by the codec, but does not specify how these bits should be packed for transmission. The packetization specified here has subsequently been adopted in ETSI Technical Specification TS 101 318. Some software implementations of the GSM codec use a different packing than that specified here.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 619
Keysight, EX-1045
Page 619

| field | field name | bits | field | field name | bits |
|---|---|---|---|---|---|
| 1 | LARc[0] | 6 | 39 | xmc[22] | 3 |
| 2 | LARc[1] | 6 | 40 | xmc[23] | 3 |
| 3 | LARc[2] | 5 | 41 | xmc[24] | 3 |
| 4 | LARc[3] | 5 | 42 | xmc[25] | 3 |
| 5 | LARc[4] | 4 | 43 | Nc[2] | 7 |
| 6 | LARc[5] | 4 | 44 | bc[2] | 2 |
| 7 | LARc[6] | 3 | 45 | Mc[2] | 2 |
| 8 | LARc[7] | 3 | 46 | xmaxc[2] | 6 |
| 9 | Nc[0] | 7 | 47 | xmc[26] | 3 |
| 10 | bc[0] | 2 | 48 | xmc[27] | 3 |
| 11 | Mc[0] | 2 | 49 | xmc[28] | 3 |
| 12 | xmaxc[0] | 6 | 50 | xmc[29] | 3 |
| 13 | xmc[0] | 3 | 51 | xmc[30] | 3 |
| 14 | xmc[1] | 3 | 52 | xmc[31] | 3 |
| 15 | xmc[2] | 3 | 53 | xmc[32] | 3 |
| 16 | xmc[3] | 3 | 54 | xmc[33] | 3 |
| 17 | xmc[4] | 3 | 55 | xmc[34] | 3 |
| 18 | xmc[5] | 3 | 56 | xmc[35] | 3 |
| 19 | xmc[6] | 3 | 57 | xmc[36] | 3 |
| 20 | xmc[7] | 3 | 58 | xmc[37] | 3 |
| 21 | xmc[8] | 3 | 59 | xmc[38] | 3 |
| 22 | xmc[9] | 3 | 60 | Nc[3] | 7 |
| 23 | xmc[10] | 3 | 61 | bc[3] | 2 |
| 24 | xmc[11] | 3 | 62 | Mc[3] | 2 |
| 25 | xmc[12] | 3 | 63 | xmaxc[3] | 6 |
| 26 | Nc[1] | 7 | 64 | xmc[39] | 3 |
| 27 | bc[1] | 2 | 65 | xmc[40] | 3 |
| 28 | Mc[1] | 2 | 66 | xmc[41] | 3 |
| 29 | xmaxc[1] | 6 | 67 | xmc[42] | 3 |
| 30 | xmc[13] | 3 | 68 | xmc[43] | 3 |
| 31 | xmc[14] | 3 | 69 | xmc[44] | 3 |
| 32 | xmc[15] | 3 | 70 | xmc[45] | 3 |
| 33 | xmc[16] | 3 | 71 | xmc[46] | 3 |
| 34 | xmc[17] | 3 | 72 | xmc[47] | 3 |
| 35 | xmc[18] | 3 | 73 | xmc[48] | 3 |
| 36 | xmc[19] | 3 | 74 | xmc[49] | 3 |
| 37 | xmc[20] | 3 | 75 | xmc[50] | 3 |
| 38 | xmc[21] | 3 | 76 | xmc[51] | 3 |

Table 2: Ordering of GSM variables



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 620
Keysight, EX-1045
Page 620

| Octet | Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | LARc0.0 | LARc0.1 | LARc0.2 | LARc0.3 |
| 1 | LARc0.4 | LARc0.5 | LARc1.0 | LARc1.1 | LARc1.2 | LARc1.3 | LARc1.4 | LARc1.5 |
| 2 | LARc2.0 | LARc2.1 | LARc2.2 | LARc2.3 | LARc2.4 | LARc3.0 | LARc3.1 | LARc3.2 |
| 3 | LARc3.3 | LARc3.4 | LARc4.0 | LARc4.1 | LARc4.2 | LARc4.3 | LARc5.0 | LARc5.1 |
| 4 | LARc5.2 | LARc5.3 | LARc6.0 | LARc6.1 | LARc6.2 | LARc7.0 | LARc7.1 | LARc7.2 |
| 5 | Nc0.0 | Nc0.1 | Nc0.2 | Nc0.3 | Nc0.4 | Nc0.5 | Nc0.6 | bc0.0 |
| 6 | bc0.1 | Mc0.0 | Mc0.1 | xmaxc00 | xmaxc01 | xmaxc02 | xmaxc03 | xmaxc04 |
| 7 | xmaxc05 | xmc0.0 | xmc0.1 | xmc0.2 | xmc1.0 | xmc1.1 | xmc1.2 | xmc2.0 |
| 8 | xmc2.1 | xmc2.2 | xmc3.0 | xmc3.1 | xmc3.2 | xmc4.0 | xmc4.1 | xmc4.2 |
| 9 | xmc5.0 | xmc5.1 | xmc5.2 | xmc6.0 | xmc6.1 | xmc6.2 | xmc7.0 | xmc7.1 |
| 10 | xmc7.2 | xmc8.0 | xmc8.1 | xmc8.2 | xmc9.0 | xmc9.1 | xmc9.2 | xmc10.0 |
| 11 | xmc10.1 | xmc10.2 | xmc11.0 | xmc11.1 | xmc11.2 | xmc12.0 | xmc12.1 | xcm12.2 |
| 12 | Nc1.0 | Nc1.1 | Nc1.2 | Nc1.3 | Nc1.4 | Nc1.5 | Nc1.6 | bc1.0 |
| 13 | bc1.1 | Mc1.0 | Mc1.1 | xmaxc10 | xmaxc11 | xmaxc12 | xmaxc13 | xmaxc14 |
| 14 | xmax15 | xmc13.0 | xmc13.1 | xmc13.2 | xmc14.0 | xmc14.1 | xmc14.2 | xmc15.0 |
| 15 | xmc15.1 | xmc15.2 | xmc16.0 | xmc16.1 | xmc16.2 | xmc17.0 | xmc17.1 | xmc17.2 |
| 16 | xmc18.0 | xmc18.1 | xmc18.2 | xmc19.0 | xmc19.1 | xmc19.2 | xmc20.0 | xmc20.1 |
| 17 | xmc20.2 | xmc21.0 | xmc21.1 | xmc21.2 | xmc22.0 | xmc22.1 | xmc22.2 | xmc23.0 |
| 18 | xmc23.1 | xmc23.2 | xmc24.0 | xmc24.1 | xmc24.2 | xmc25.0 | xmc25.1 | xmc25.2 |
| 19 | Nc2.0 | Nc2.1 | Nc2.2 | Nc2.3 | Nc2.4 | Nc2.5 | Nc2.6 | bc2.0 |
| 20 | bc2.1 | Mc2.0 | Mc2.1 | xmaxc20 | xmaxc21 | xmaxc22 | xmaxc23 | xmaxc24 |
| 21 | xmaxc25 | xmc26.0 | xmc26.1 | xmc26.2 | xmc27.0 | xmc27.1 | xmc27.2 | xmc28.0 |
| 22 | xmc28.1 | xmc28.2 | xmc29.0 | xmc29.1 | xmc29.2 | xmc30.0 | xmc30.1 | xmc30.2 |
| 23 | xmc31.0 | xmc31.1 | xmc31.2 | xmc32.0 | xmc32.1 | xmc32.2 | xmc33.0 | xmc33.1 |
| 24 | xmc33.2 | xmc34.0 | xmc34.1 | xmc34.2 | xmc35.0 | xmc35.1 | xmc35.2 | xmc36.0 |
| 25 | Xmc36.1 | xmc36.2 | xmc37.0 | xmc37.1 | xmc37.2 | xmc38.0 | xmc38.1 | xmc38.2 |
| 26 | Nc3.0 | Nc3.1 | Nc3.2 | Nc3.3 | Nc3.4 | Nc3.5 | Nc3.6 | bc3.0 |
| 27 | bc3.1 | Mc3.0 | Mc3.1 | xmaxc30 | xmaxc31 | xmaxc32 | xmaxc33 | xmaxc34 |
| 28 | xmaxc35 | xmc39.0 | xmc39.1 | xmc39.2 | xmc40.0 | xmc40.1 | xmc40.2 | xmc41.0 |
| 29 | xmc41.1 | xmc41.2 | xmc42.0 | xmc42.1 | xmc42.2 | xmc43.0 | xmc43.1 | xmc43.2 |
| 30 | xmc44.0 | xmc44.1 | xmc44.2 | xmc45.0 | xmc45.1 | xmc45.2 | xmc46.0 | xmc46.1 |
| 31 | xmc46.2 | xmc47.0 | xmc47.1 | xmc47.2 | xmc48.0 | xmc48.1 | xmc48.2 | xmc49.0 |
| 32 | xmc49.1 | xmc49.2 | xmc50.0 | xmc50.1 | xmc50.2 | xmc51.0 | xmc51.1 | xmc51.2 |

Table 3: GSM payload format

In the GSM packing used by RTP, the bits SHALL be packed beginning from the most significant bit. Every 160 sample GSM frame is coded into one 33 octet (264 bit) buffer. Every such buffer begins with a 4 bit signature (0xD), followed by the MSB encoding of the fields of the frame. The first octet thus contains 1101 in the 4 most significant bits (0-3) and the 4 most significant bits of F1 (0-3) in the 4 least significant bits (4-7). The second octet contains the 2 least significant bits of F1 in bits 0-1, and F2 in bits 2-7, and so on. The order of the fields in the frame is described in Table 2.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 621
Keysight, EX-1045
Page 621

#### 4.5.8.2 GSM Variable Names and Numbers

In the RTP encoding we have the bit pattern described in Table 3, where F.i signifies the ith bit of the field F, bit 0 is the most significant bit, and the bits of every octet are numbered from 0 to 7 from most to least significant.

### 4.5.9 GSM-EFR

GSM-EFR denotes GSM 06.60 enhanced full rate speech transcoding, specified in ETS 300 726 which is available from ETSI at the address given in Section 4.5.8. This codec has a frame length of 244 bits. For transmission in RTP, each codec frame is packed into a 31 octet (248 bit) buffer beginning with a 4-bit signature 0xC in a manner similar to that specified here for the original GSM 06.10 codec. The packing is specified in ETSI Technical Specification TS 101 318.

### 4.5.10 L8

L8 denotes linear audio data samples, using 8-bits of precision with an offset of 128, that is, the most negative signal is encoded as zero.

### 4.5.11 L16

L16 denotes uncompressed audio data samples, using 16-bit signed representation with 65,535 equally divided steps between minimum and maximum signal level, ranging from -32,768 to 32,767. The value is represented in two's complement notation and transmitted in network byte order (most significant byte first).

The MIME registration for L16 in RFC 3555 [7] specifies parameters that MAY be used with MIME or SDP to indicate that analog pre-emphasis was applied to the signal before quantization or to indicate that a multiple-channel audio stream follows a different channel ordering convention than is specified in Section 4.1.

### 4.5.12 LPC

LPC designates an experimental linear predictive encoding contributed by Ron Frederick, which is based on an implementation written by Ron Zuckerman posted to the Usenet group comp.dsp on June 26, 1992. The codec generates 14 octets for every frame. The framesize is set to 20 ms, resulting in a bit rate of 5,600 b/s.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 622
Keysight, EX-1045
Page 622

### 4.5.13 MPA

MPA denotes MPEG-1 or MPEG-2 audio encapsulated as elementary streams. The encoding is defined in ISO standards ISO/IEC 11172-3 and 13818-3. The encapsulation is specified in RFC 2250 [14].

The encoding may be at any of three levels of complexity, called Layer I, II and III. The selected layer as well as the sampling rate and channel count are indicated in the payload. The RTP timestamp clock rate is always 90,000, independent of the sampling rate. MPEG-1 audio supports sampling rates of 32, 44.1, and 48 kHz (ISO/IEC 11172-3, section 1.1; "Scope"). MPEG-2 supports sampling rates of 16, 22.05 and 24 kHz. The number of samples per frame is fixed, but the frame size will vary with the sampling rate and bit rate.

The MIME registration for MPA in RFC 3555 [7] specifies parameters that MAY be used with MIME or SDP to restrict the selection of layer, channel count, sampling rate, and bit rate.

### 4.5.14 PCMA and PCMU

PCMA and PCMU are specified in ITU-T Recommendation G.711. Audio data is encoded as eight bits per sample, after logarithmic scaling. PCMU denotes mu-law scaling, PCMA A-law scaling. A detailed description is given by Jayant and Noll [15]. Each G.711 octet SHALL be octet-aligned in an RTP packet. The sign bit of each G.711 octet SHALL correspond to the most significant bit of the octet in the RTP packet (i.e., assuming the G.711 samples are handled as octets on the host machine, the sign bit SHALL be the most significant bit of the octet as defined by the host machine format). The 56 kb/s and 48 kb/s modes of G.711 are not applicable to RTP, since PCMA and PCMU MUST always be transmitted as 8-bit samples.

See Section 4.1 regarding silence suppression.

### 4.5.15 QCELP

The Electronic Industries Association (EIA) & Telecommunications Industry Association (TIA) standard IS-733, "TR45: High Rate Speech Service Option for Wideband Spread Spectrum Communications Systems", defines the QCELP audio compression algorithm for use in wireless CDMA applications. The QCELP CODEC compresses each 20 milliseconds of 8,000 Hz, 16-bit sampled input speech into one of four different size output frames: Rate 1 (266 bits), Rate 1/2 (124 bits), Rate 1/4 (54 bits) or Rate 1/8 (20 bits). For typical speech patterns, this results in an average output of 6.8 kb/s for normal mode and 4.7 kb/s for reduced rate mode. The packetization of the QCELP audio codec is described in [16].



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 623
Keysight, EX-1045
Page 623

### 4.5.16 RED

The redundant audio payload format "RED" is specified by RFC 2198 [17]. It defines a means by which multiple redundant copies of an audio packet may be transmitted in a single RTP stream. Each packet in such a stream contains, in addition to the audio data for that packetization interval, a (more heavily compressed) copy of the data from a previous packetization interval. This allows an approximation of the data from lost packets to be recovered upon decoding of a subsequent packet, giving much improved sound quality when compared with silence substitution for lost packets.

### 4.5.17 VDVI

VDVI is a variable-rate version of DVI4, yielding speech bit rates of between 10 and 25 kb/s. It is specified for single-channel operation only. Samples are packed into octets starting at the most-significant bit. The last octet is padded with 1 bits if the last sample does not fill the last octet. This padding is distinct from the valid codewords. The receiver needs to detect the padding because there is no explicit count of samples in the packet.

It uses the following encoding:

| DVI4 codeword | VDVI bit pattern |
|---|---|
| 0 | 00 |
| 1 | 010 |
| 2 | 1100 |
| 3 | 11100 |
| 4 | 111100 |
| 5 | 1111100 |
| 6 | 11111100 |
| 7 | 11111110 |
| 8 | 10 |
| 9 | 011 |
| 10 | 1101 |
| 11 | 11101 |
| 12 | 111101 |
| 13 | 1111101 |
| 14 | 11111101 |
| 15 | 11111111 |



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 624
Keysight, EX-1045
Page 624

## 5. Video

The following sections describe the video encodings that are defined in this memo and give their abbreviated names used for identification. These video encodings and their payload types are listed in Table 5.

All of these video encodings use an RTP timestamp frequency of 90,000 Hz, the same as the MPEG presentation time stamp frequency. This frequency yields exact integer timestamp increments for the typical 24 (HDTV), 25 (PAL), and 29.97 (NTSC) and 30 Hz (HDTV) frame rates and 50, 59.94 and 60 Hz field rates. While 90 kHz is the RECOMMENDED rate for future video encodings used within this profile, other rates MAY be used. However, it is not sufficient to use the video frame rate (typically between 15 and 30 Hz) because that does not provide adequate resolution for typical synchronization requirements when calculating the RTP timestamp corresponding to the NTP timestamp in an RTCP SR packet. The timestamp resolution MUST also be sufficient for the jitter estimate contained in the receiver reports.

For most of these video encodings, the RTP timestamp encodes the sampling instant of the video image contained in the RTP data packet. If a video image occupies more than one packet, the timestamp is the same on all of those packets. Packets from different video images are distinguished by their different timestamps.

Most of these video encodings also specify that the marker bit of the RTP header SHOULD be set to one in the last packet of a video frame and otherwise set to zero. Thus, it is not necessary to wait for a following packet with a different timestamp to detect that a new frame should be displayed.

### 5.1 CelB

The CELL-B encoding is a proprietary encoding proposed by Sun Microsystems. The byte stream format is described in RFC 2029 [18].

### 5.2 JPEG

The encoding is specified in ISO Standards 10918-1 and 10918-2. The RTP payload format is as specified in RFC 2435 [19].

### 5.3 H261

The encoding is specified in ITU-T Recommendation H.261, "Video codec for audiovisual services at p x 64 kbit/s". The packetization and RTP-specific properties are described in RFC 2032 [20].



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 625
Keysight, EX-1045
Page 625

### 5.4 H263

The encoding is specified in the 1996 version of ITU-T Recommendation H.263, "Video coding for low bit rate communication". The packetization and RTP-specific properties are described in RFC 2190 [21]. The H263-1998 payload format is RECOMMENDED over this one for use by new implementations.

### 5.5 H263-1998

The encoding is specified in the 1998 version of ITU-T Recommendation H.263, "Video coding for low bit rate communication". The packetization and RTP-specific properties are described in RFC 2429 [22]. Because the 1998 version of H.263 is a superset of the 1996 syntax, this payload format can also be used with the 1996 version of H.263, and is RECOMMENDED for this use by new implementations. This payload format does not replace RFC 2190, which continues to be used by existing implementations, and may be required for backward compatibility in new implementations. Implementations using the new features of the 1998 version of H.263 MUST use the payload format described in RFC 2429.

### 5.6 MPV

MPV designates the use of MPEG-1 and MPEG-2 video encoding elementary streams as specified in ISO Standards ISO/IEC 11172 and 13818-2, respectively. The RTP payload format is as specified in RFC 2250 [14], Section 3.

The MIME registration for MPV in RFC 3555 [7] specifies a parameter that MAY be used with MIME or SDP to restrict the selection of the type of MPEG video.

### 5.7 MP2T

MP2T designates the use of MPEG-2 transport streams, for either audio or video. The RTP payload format is described in RFC 2250 [14], Section 2.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 626
Keysight, EX-1045
Page 626

### 5.8 nv

The encoding is implemented in the program `nv', version 4, developed at Xerox PARC by Ron Frederick. Further information is available from the author:

Ron Frederick
Blue Coat Systems Inc.
650 Almanor Avenue
Sunnyvale, CA 94085
United States
EMail: ronf@bluecoat.com

## 6. Payload Type Definitions

Tables 4 and 5 define this profile's static payload type values for the PT field of the RTP data header. In addition, payload type values in the range 96-127 MAY be defined dynamically through a conference control protocol, which is beyond the scope of this document. For example, a session directory could specify that for a given session, payload type 96 indicates PCMU encoding, 8,000 Hz sampling rate, 2 channels. Entries in Tables 4 and 5 with payload type "dyn" have no static payload type assigned and are only used with a dynamic payload type. Payload type 2 was assigned to G721 in RFC 1890 and to its equivalent successor G726-32 in draft versions of this specification, but its use is now deprecated and that static payload type is marked reserved due to conflicting use for the payload formats G726-32 and AAL2-G726-32 (see Section 4.5.4). Payload type 13 indicates the Comfort Noise (CN) payload format specified in RFC 3389 [9]. Payload type 19 is marked "reserved" because some draft versions of this specification assigned that number to an earlier version of the comfort noise payload format. The payload type range 72-76 is marked "reserved" so that RTCP and RTP packets can be reliably distinguished (see Section "Summary of Protocol Constants" of the RTP protocol specification).

The payload types currently defined in this profile are assigned to exactly one of three categories or media types: audio only, video only and those combining audio and video. The media types are marked in Tables 4 and 5 as "A", "V" and "AV", respectively. Payload types of different media types SHALL NOT be interleaved or multiplexed within a single RTP session, but multiple RTP sessions MAY be used in parallel to send multiple media types. An RTP source MAY change payload types within the same media type during a session. See the section "Multiplexing RTP Sessions" of RFC 3550 for additional explanation.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 627
Keysight, EX-1045
Page 627

| PT | encoding name | media type | clock rate (Hz) | channels |
|---|---|---|---|---|
| 0 | PCMU | A | 8,000 | 1 |
| 1 | reserved | A | | |
| 2 | reserved | A | | |
| 3 | GSM | A | 8,000 | 1 |
| 4 | G723 | A | 8,000 | 1 |
| 5 | DVI4 | A | 8,000 | 1 |
| 6 | DVI4 | A | 16,000 | 1 |
| 7 | LPC | A | 8,000 | 1 |
| 8 | PCMA | A | 8,000 | 1 |
| 9 | G722 | A | 8,000 | 1 |
| 10 | L16 | A | 44,100 | 2 |
| 11 | L16 | A | 44,100 | 1 |
| 12 | QCELP | A | 8,000 | 1 |
| 13 | CN | A | 8,000 | 1 |
| 14 | MPA | A | 90,000 | (see text) |
| 15 | G728 | A | 8,000 | 1 |
| 16 | DVI4 | A | 11,025 | 1 |
| 17 | DVI4 | A | 22,050 | 1 |
| 18 | G729 | A | 8,000 | 1 |
| 19 | reserved | A | | |
| 20 | unassigned | A | | |
| 21 | unassigned | A | | |
| 22 | unassigned | A | | |
| 23 | unassigned | A | | |
| dyn | G726-40 | A | 8,000 | 1 |
| dyn | G726-32 | A | 8,000 | 1 |
| dyn | G726-24 | A | 8,000 | 1 |
| dyn | G726-16 | A | 8,000 | 1 |
| dyn | G729D | A | 8,000 | 1 |
| dyn | G729E | A | 8,000 | 1 |
| dyn | GSM-EFR | A | 8,000 | 1 |
| dyn | L8 | A | var. | var. |
| dyn | RED | A | | (see text) |
| dyn | VDVI | A | var. | 1 |

Table 4: Payload types (PT) for audio encodings



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 628
Keysight, EX-1045
Page 628

| PT | encoding name | media type | clock rate (Hz) |
|---|---|---|---|
| 24 | unassigned | V | |
| 25 | CelB | V | 90,000 |
| 26 | JPEG | V | 90,000 |
| 27 | unassigned | V | |
| 28 | nv | V | 90,000 |
| 29 | unassigned | V | |
| 30 | unassigned | V | |
| 31 | H261 | V | 90,000 |
| 32 | MPV | V | 90,000 |
| 33 | MP2T | AV | 90,000 |
| 34 | H263 | V | 90,000 |
| 35-71 | unassigned | ? | |
| 72-76 | reserved | N/A | N/A |
| 77-95 | unassigned | ? | |
| 96-127 | dynamic | ? | |
| dyn | H263-1998 | V | 90,000 |

Table 5: Payload types (PT) for video and combined encodings

Session participants agree through mechanisms beyond the scope of this specification on the set of payload types allowed in a given session. This set MAY, for example, be defined by the capabilities of the applications used, negotiated by a conference control protocol or established by agreement between the human participants.

Audio applications operating under this profile SHOULD, at a minimum, be able to send and/or receive payload types 0 (PCMU) and 5 (DVI4). This allows interoperability without format negotiation and ensures successful negotiation with a conference control protocol.

## 7. RTP over TCP and Similar Byte Stream Protocols

Under special circumstances, it may be necessary to carry RTP in protocols offering a byte stream abstraction, such as TCP, possibly multiplexed with other data. The application MUST define its own method of delineating RTP and RTCP packets (RTSP [23] provides an example of such an encapsulation specification).

## 8. Port Assignment

As specified in the RTP protocol definition, RTP data SHOULD be carried on an even UDP port number and the corresponding RTCP packets SHOULD be carried on the next higher (odd) port number.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 629
Keysight, EX-1045
Page 629

Applications operating under this profile MAY use any such UDP port pair. For example, the port pair MAY be allocated randomly by a session management program. A single fixed port number pair cannot be required because multiple applications using this profile are likely to run on the same host, and there are some operating systems that do not allow multiple processes to use the same UDP port with different multicast addresses.

However, port numbers 5004 and 5005 have been registered for use with this profile for those applications that choose to use them as the default pair. Applications that operate under multiple profiles MAY use this port pair as an indication to select this profile if they are not subject to the constraint of the previous paragraph. Applications need not have a default and MAY require that the port pair be explicitly specified. The particular port numbers were chosen to lie in the range above 5000 to accommodate port number allocation practice within some versions of the Unix operating system, where port numbers below 1024 can only be used by privileged processes and port numbers between 1024 and 5000 are automatically assigned by the operating system.

## 9. Changes from RFC 1890

This RFC revises RFC 1890. It is mostly backwards-compatible with RFC 1890 except for functions removed because two interoperable implementations were not found. The additions to RFC 1890 codify existing practice in the use of payload formats under this profile. Since this profile may be used without using any of the payload formats listed here, the addition of new payload formats in this revision does not affect backwards compatibility. The changes are listed below, categorized into functional and non-functional changes.

Functional changes:

- Section 11, "IANA Considerations" was added to specify the registration of the name for this profile. That appendix also references a new Section 3 "Registering Additional Encodings" which establishes a policy that no additional registration of static payload types for this profile will be made beyond those added in this revision and included in Tables 4 and 5. Instead, additional encoding names may be registered as MIME subtypes for binding to dynamic payload types. Non-normative references were added to RFC 3555 [7] where MIME subtypes for all the listed payload formats are registered, some with optional parameters for use of the payload formats.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 630
Keysight, EX-1045
Page 630

- Static payload types 4, 16, 17 and 34 were added to incorporate IANA registrations made since the publication of RFC 1890, along with the corresponding payload format descriptions for G723 and H263.

- Following working group discussion, static payload types 12 and 18 were added along with the corresponding payload format descriptions for QCELP and G729. Static payload type 13 was assigned to the Comfort Noise (CN) payload format defined in RFC 3389. Payload type 19 was marked reserved because it had been temporarily allocated to an earlier version of Comfort Noise present in some draft revisions of this document.

- The payload format for G721 was renamed to G726-32 following the ITU-T renumbering, and the payload format description for G726 was expanded to include the -16, -24 and -40 data rates. Because of confusion regarding draft revisions of this document, some implementations of these G726 payload formats packed samples into octets starting with the most significant bit rather than the least significant bit as specified here. To partially resolve this incompatibility, new payload formats named AAL2-G726-16, -24, -32 and -40 will be specified in a separate document (see note in Section 4.5.4), and use of static payload type 2 is deprecated as explained in Section 6.

- Payload formats G729D and G729E were added following the ITU-T addition of Annexes D and E to Recommendation G.729. Listings were added for payload formats GSM-EFR, RED, and H263-1998 published in other documents subsequent to RFC 1890. These additional payload formats are referenced only by dynamic payload type numbers.

- The descriptions of the payload formats for G722, G728, GSM, VDVI were expanded.

- The payload format for 1016 audio was removed and its static payload type assignment 1 was marked "reserved" because two interoperable implementations were not found.

- Requirements for congestion control were added in Section 2.

- This profile follows the suggestion in the revised RTP spec that RTCP bandwidth may be specified separately from the session bandwidth and separately for active senders and passive receivers.

- The mapping of a user pass-phrase string into an encryption key was deleted from Section 2 because two interoperable implementations were not found.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 631
Keysight, EX-1045
Page 631

- The "quadrophonic" sample ordering convention for four-channel audio was removed to eliminate an ambiguity as noted in Section 4.1.

Non-functional changes:

- In Section 4.1, it is now explicitly stated that silence suppression is allowed for all audio payload formats. (This has always been the case and derives from a fundamental aspect of RTP's design and the motivations for packet audio, but was not explicit stated before.) The use of comfort noise is also explained.

- In Section 4.1, the requirement level for setting of the marker bit on the first packet after silence for audio was changed from "is" to "SHOULD be", and clarified that the marker bit is set only when packets are intentionally not sent.

- Similarly, text was added to specify that the marker bit SHOULD be set to one on the last packet of a video frame, and that video frames are distinguished by their timestamps.

- RFC references are added for payload formats published after RFC 1890.

- The security considerations and full copyright sections were added.

- According to Peter Hoddie of Apple, only pre-1994 Macintosh used the 22254.54 rate and none the 11127.27 rate, so the latter was dropped from the discussion of suggested sampling frequencies.

- Table 1 was corrected to move some values from the "ms/packet" column to the "default ms/packet" column where they belonged.

- Since the Interactive Multimedia Association ceased operations, an alternate resource was provided for a referenced IMA document.

- A note has been added for G722 to clarify a discrepancy between the actual sampling rate and the RTP timestamp clock rate.

- Small clarifications of the text have been made in several places, some in response to questions from readers. In particular:

  - A definition for "media type" is given in Section 1.1 to allow the explanation of multiplexing RTP sessions in Section 6 to be more clear regarding the multiplexing of multiple media.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 632
Keysight, EX-1045
Page 632

- The explanation of how to determine the number of audio frames in a packet from the length was expanded.

- More description of the allocation of bandwidth to SDES items is given.

- A note was added that the convention for the order of channels specified in Section 4.1 may be overridden by a particular encoding or payload format specification.

- The terms MUST, SHOULD, MAY, etc. are used as defined in RFC 2119.

o A second author for this document was added.

## 10. Security Considerations

Implementations using the profile defined in this specification are subject to the security considerations discussed in the RTP specification [1]. This profile does not specify any different security services. The primary function of this profile is to list a set of data compression encodings for audio and video media.

Confidentiality of the media streams is achieved by encryption. Because the data compression used with the payload formats described in this profile is applied end-to-end, encryption may be performed after compression so there is no conflict between the two operations.

A potential denial-of-service threat exists for data encodings using compression techniques that have non-uniform receiver-end computational load. The attacker can inject pathological datagrams into the stream which are complex to decode and cause the receiver to be overloaded.

As with any IP-based protocol, in some circumstances a receiver may be overloaded simply by the receipt of too many packets, either desired or undesired. Network-layer authentication MAY be used to discard packets from undesired sources, but the processing cost of the authentication itself may be too high. In a multicast environment, source pruning is implemented in IGMPv3 (RFC 3376) [24] and in multicast routing protocols to allow a receiver to select which sources are allowed to reach it.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 633
Keysight, EX-1045
Page 633

## 11. IANA Considerations

The RTP specification establishes a registry of profile names for use by higher-level control protocols, such as the Session Description Protocol (SDP), RFC 2327 [6], to refer to transport methods. This profile registers the name "RTP/AVP".

Section 3 establishes the policy that no additional registration of static RTP payload types for this profile will be made beyond those added in this document revision and included in Tables 4 and 5. IANA may reference that section in declining to accept any additional registration requests. In Tables 4 and 5, note that types 1 and 2 have been marked reserved and the set of "dyn" payload types included has been updated. These changes are explained in Sections 6 and 9.

## 12. References

### 12.1 Normative References

[1] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.

[2] Bradner, S., "Key Words for Use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[3] Apple Computer, "Audio Interchange File Format AIFF-C", August 1991. (also ftp://ftp.sgi.com/sgi/aiff-c.9.26.91.ps.Z).

### 12.2 Informative References

[4] Braden, R., Clark, D. and S. Shenker, "Integrated Services in the Internet Architecture: an Overview", RFC 1633, June 1994.

[5] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z. and W. Weiss, "An Architecture for Differentiated Service", RFC 2475, December 1998.

[6] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

[7] Casner, S. and P. Hoschka, "MIME Type Registration of RTP Payload Types", RFC 3555, July 2003.

[8] Freed, N., Klensin, J. and J. Postel, "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures", BCP 13, RFC 2048, November 1996.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 634
Keysight, EX-1045
Page 634

[9] Zopf, R., "Real-time Transport Protocol (RTP) Payload for Comfort Noise (CN)", RFC 3389, September 2002.

[10] Deleam, D. and J.-P. Petit, "Real-time implementations of the recent ITU-T low bit rate speech coders on the TI TMS320C54X DSP: results, methodology, and applications", in Proc. of International Conference on Signal Processing, Technology, and Applications (ICSPAT) , (Boston, Massachusetts), pp. 1656--1660, October 1996.

[11] Mouly, M. and M.-B. Pautet, The GSM system for mobile communications Lassay-les-Chateaux, France: Europe Media Duplication, 1993.

[12] Degener, J., "Digital Speech Compression", Dr. Dobb's Journal, December 1994.

[13] Redl, S., Weber, M. and M. Oliphant, An Introduction to GSM Boston: Artech House, 1995.

[14] Hoffman, D., Fernando, G., Goyal, V. and M. Civanlar, "RTP Payload Format for MPEG1/MPEG2 Video", RFC 2250, January 1998.

[15] Jayant, N. and P. Noll, Digital Coding of Waveforms--Principles and Applications to Speech and Video Englewood Cliffs, New Jersey: Prentice-Hall, 1984.

[16] McKay, K., "RTP Payload Format for PureVoice(tm) Audio", RFC 2658, August 1999.

[17] Perkins, C., Kouvelas, I., Hodson, O., Hardman, V., Handley, M., Bolot, J.-C., Vega-Garcia, A. and S. Fosse-Parisis, "RTP Payload for Redundant Audio Data", RFC 2198, September 1997.

[18] Speer, M. and D. Hoffman, "RTP Payload Format of Sun's CellB Video Encoding", RFC 2029, October 1996.

[19] Berc, L., Fenner, W., Frederick, R., McCanne, S. and P. Stewart, "RTP Payload Format for JPEG-Compressed Video", RFC 2435, October 1998.

[20] Turletti, T. and C. Huitema, "RTP Payload Format for H.261 Video Streams", RFC 2032, October 1996.

[21] Zhu, C., "RTP Payload Format for H.263 Video Streams", RFC 2190, September 1997.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 635
**Keysight, EX-1045**
**Page 635**

[22] Bormann, C., Cline, L., Deisher, G., Gardos, T., Maciocco, C., Newell, D., Ott, J., Sullivan, G., Wenger, S. and C. Zhu, "RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+)", RFC 2429, October 1998.

[23] Schulzrinne, H., Rao, A. and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

[24] Cain, B., Deering, S., Kouvelas, I., Fenner, B. and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

## 13. Current Locations of Related Resources

Note: Several sections below refer to the ITU-T Software Tool Library (STL). It is available from the ITU Sales Service, Place des Nations, CH-1211 Geneve 20, Switzerland (also check http://www.itu.int). The ITU-T STL is covered by a license defined in ITU-T Recommendation G.191, "Software tools for speech and audio coding standardization".

DVI4

An archived copy of the document IMA Recommended Practices for Enhancing Digital Audio Compatibility in Multimedia Systems (version 3.0), which describes the IMA ADPCM algorithm, is available at:

http://www.cs.columbia.edu/~hgs/audio/dvi/

An implementation is available from Jack Jansen at

ftp://ftp.cwi.nl/local/pub/audio/adpcm.shar

G722

An implementation of the G.722 algorithm is available as part of the ITU-T STL, described above.

G723

The reference C code implementation defining the G.723.1 algorithm and its Annexes A, B, and C are available as an integral part of Recommendation G.723.1 from the ITU Sales Service, address listed above. Both the algorithm and C code are covered by a specific license. The ITU-T Secretariat should be contacted to obtain such licensing information.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 636
Keysight, EX-1045
Page 636

G726

G726 is specified in the ITU-T Recommendation G.726, "40, 32, 24, and 16 kb/s Adaptive Differential Pulse Code Modulation (ADPCM)". An implementation of the G.726 algorithm is available as part of the ITU-T STL, described above.

G729

The reference C code implementation defining the G.729 algorithm and its Annexes A through I are available as an integral part of Recommendation G.729 from the ITU Sales Service, listed above. Annex I contains the integrated C source code for all G.729 operating modes. The G.729 algorithm and associated C code are covered by a specific license. The contact information for obtaining the license is available from the ITU-T Secretariat.

GSM

A reference implementation was written by Carsten Bormann and Jutta Degener (then at TU Berlin, Germany). It is available at

http://www.dmn.tzi.org/software/gsm/

Although the RPE-LTP algorithm is not an ITU-T standard, there is a C code implementation of the RPE-LTP algorithm available as part of the ITU-T STL. The STL implementation is an adaptation of the TU Berlin version.

LPC

An implementation is available at

ftp://parcftp.xerox.com/pub/net-research/lpc.tar.Z

PCMU, PCMA

An implementation of these algorithms is available as part of the ITU-T STL, described above.

## 14. Acknowledgments

The comments and careful review of Simao Campos, Richard Cox and AVT Working Group participants are gratefully acknowledged. The GSM description was adopted from the IMTC Voice over IP Forum Service Interoperability Implementation Agreement (January 1997). Fred Burg and Terry Lyons helped with the G.729 description.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 637
Keysight, EX-1045
Page 637

## 15. Intellectual Property Rights Statement

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

## 16. Authors' Addresses

Henning Schulzrinne
Department of Computer Science
Columbia University
1214 Amsterdam Avenue
New York, NY 10027
United States

EMail: schulzrinne@cs.columbia.edu

Stephen L. Casner
Packet Design
3400 Hillview Avenue, Building 3
Palo Alto, CA 94304
United States

EMail: casner@acm.org



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 638
Keysight, EX-1045
Page 638

## 17. Full Copyright Statement

Copyright (C) The Internet Society (2003). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.



https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 639
Keysight, EX-1045
Page 639

https://web.archive.org/web/20070206011733/http://tools.ietf.org/html/rfc3551

Updated by: rfc3265 rfc3853 rfc4320 Errata

Network Working Group
Request for Comments: 3261
Obsoletes: 2543
Category: Standards Track

J. Rosenberg
dynamicsoft
H. Schulzrinne
Columbia U.
G. Camarillo
Ericsson
A. Johnston
WorldCom
J. Peterson
Neustar
R. Sparks
dynamicsoft
M. Handley
ICIR
E. Schooler
AT&T

June 2002

# SIP: Session Initiation Protocol

## Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Copyright Notice

Copyright (C) The Internet Society (2002). All Rights Reserved.

## Abstract

This document describes Session Initiation Protocol (SIP), an application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. These sessions include Internet telephone calls, multimedia distribution, and multimedia conferences.

SIP invitations used to create sessions carry session descriptions that allow participants to agree on a set of compatible media types. SIP makes use of elements called proxy servers to help route requests to the user's current location, authenticate and authorize users for services, implement provider call-routing policies, and provide features to users. SIP also provides a registration function that allows users to upload their current locations for use by proxy servers. SIP runs on top of several different transport protocols.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 641
Keysight, EX-1045
Page 641

Table of Contents


1 Introduction ..... 8
2 Overview of SIP Functionality ..... 9
3 Terminology ..... 10
4 Overview of Operation ..... 10
5 Structure of the Protocol ..... 18
6 Definitions ..... 20
7 SIP Messages ..... 26
7.1 Requests ..... 27
7.2 Responses ..... 28
7.3 Header Fields ..... 29
7.3.1 Header Field Format ..... 30
7.3.2 Header Field Classification ..... 32
7.3.3 Compact Form ..... 32
7.4 Bodies ..... 33
7.4.1 Message Body Type ..... 33
7.4.2 Message Body Length ..... 33
7.5 Framing SIP Messages ..... 34
8 General User Agent Behavior ..... 34
8.1 UAC Behavior ..... 35
8.1.1 Generating the Request ..... 35
8.1.1.1 Request-URI ..... 35
8.1.1.2 To ..... 36
8.1.1.3 From ..... 37
8.1.1.4 Call-ID ..... 37
8.1.1.5 CSeq ..... 38
8.1.1.6 Max-Forwards ..... 38
8.1.1.7 Via ..... 39
8.1.1.8 Contact ..... 40
8.1.1.9 Supported and Require ..... 40
8.1.1.10 Additional Message Components ..... 41
8.1.2 Sending the Request ..... 41
8.1.3 Processing Responses ..... 42
8.1.3.1 Transaction Layer Errors ..... 42
8.1.3.2 Unrecognized Responses ..... 42
8.1.3.3 Vias ..... 43
8.1.3.4 Processing 3xx Responses ..... 43
8.1.3.5 Processing 4xx Responses ..... 45
8.2 UAS Behavior ..... 46
8.2.1 Method Inspection ..... 46
8.2.2 Header Inspection ..... 46
8.2.2.1 To and Request-URI ..... 46
8.2.2.2 Merged Requests ..... 47
8.2.2.3 Require ..... 47
8.2.3 Content Processing ..... 48
8.2.4 Applying Extensions ..... 49
8.2.5 Processing the Request ..... 49







8.2.6 Generating the Response ..... 49
8.2.6.1 Sending a Provisional Response ..... 49
8.2.6.2 Headers and Tags ..... 50
8.2.7 Stateless UAS Behavior ..... 50
8.3 Redirect Servers ..... 51
9 Canceling a Request ..... 53


https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 642
Keysight, EX-1045
Page 642

9.1 Client Behavior ..... 53
9.2 Server Behavior ..... 55
10 Registrations ..... 56
10.1 Overview ..... 56
10.2 Constructing the REGISTER Request ..... 57
10.2.1 Adding Bindings ..... 59
10.2.1.1 Setting the Expiration Interval of Contact Addresses 60
10.2.1.2 Preferences among Contact Addresses ..... 61
10.2.2 Removing Bindings ..... 61
10.2.3 Fetching Bindings ..... 61
10.2.4 Refreshing Bindings ..... 61
10.2.5 Setting the Internal Clock ..... 62
10.2.6 Discovering a Registrar ..... 62
10.2.7 Transmitting a Request ..... 62
10.2.8 Error Responses ..... 63
10.3 Processing REGISTER Requests ..... 63
11 Querying for Capabilities ..... 66
11.1 Construction of OPTIONS Request ..... 67
11.2 Processing of OPTIONS Request ..... 68
12 Dialogs ..... 69
12.1 Creation of a Dialog ..... 70
12.1.1 UAS behavior ..... 70
12.1.2 UAC Behavior ..... 71
12.2 Requests within a Dialog ..... 72
12.2.1 UAC Behavior ..... 73
12.2.1.1 Generating the Request ..... 73
12.2.1.2 Processing the Responses ..... 75
12.2.2 UAS Behavior ..... 76
12.3 Termination of a Dialog ..... 77
13 Initiating a Session ..... 77
13.1 Overview ..... 77
13.2 UAC Processing ..... 78
13.2.1 Creating the Initial INVITE ..... 78
13.2.2 Processing INVITE Responses ..... 81
13.2.2.1 1xx Responses ..... 81
13.2.2.2 3xx Responses ..... 81
13.2.2.3 4xx, 5xx and 6xx Responses ..... 81
13.2.2.4 2xx Responses ..... 82
13.3 UAS Processing ..... 83
13.3.1 Processing of the INVITE ..... 83
13.3.1.1 Progress ..... 84
13.3.1.2 The INVITE is Redirected ..... 84







13.3.1.3 The INVITE is Rejected ..... 85
13.3.1.4 The INVITE is Accepted ..... 85
14 Modifying an Existing Session ..... 86
14.1 UAC Behavior ..... 86
14.2 UAS Behavior ..... 88
15 Terminating a Session ..... 89
15.1 Terminating a Session with a BYE Request ..... 90
15.1.1 UAC Behavior ..... 90
15.1.2 UAS Behavior ..... 91
16 Proxy Behavior ..... 91
16.1 Overview ..... 91
16.2 Stateful Proxy ..... 92
16.3 Request Validation ..... 94


https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 643
Keysight, EX-1045
Page 643

16.4 Route Information Preprocessing ..... 96
16.5 Determining Request Targets ..... 97
16.6 Request Forwarding ..... 99
16.7 Response Processing ..... 107
16.8 Processing Timer C ..... 114
16.9 Handling Transport Errors ..... 115
16.10 CANCEL Processing ..... 115
16.11 Stateless Proxy ..... 116
16.12 Summary of Proxy Route Processing ..... 118
16.12.1 Examples ..... 118
16.12.1.1 Basic SIP Trapezoid ..... 118
16.12.1.2 Traversing a Strict-Routing Proxy ..... 120
16.12.1.3 Rewriting Record-Route Header Field Values ..... 121
17 Transactions ..... 122
17.1 Client Transaction ..... 124
17.1.1 INVITE Client Transaction ..... 125
17.1.1.1 Overview of INVITE Transaction ..... 125
17.1.1.2 Formal Description ..... 125
17.1.1.3 Construction of the ACK Request ..... 129
17.1.2 Non-INVITE Client Transaction ..... 130
17.1.2.1 Overview of the non-INVITE Transaction ..... 130
17.1.2.2 Formal Description ..... 131
17.1.3 Matching Responses to Client Transactions ..... 132
17.1.4 Handling Transport Errors ..... 133
17.2 Server Transaction ..... 134
17.2.1 INVITE Server Transaction ..... 134
17.2.2 Non-INVITE Server Transaction ..... 137
17.2.3 Matching Requests to Server Transactions ..... 138
17.2.4 Handling Transport Errors ..... 141
18 Transport ..... 141
18.1 Clients ..... 142
18.1.1 Sending Requests ..... 142
18.1.2 Receiving Responses ..... 144
18.2 Servers ..... 145
18.2.1 Receiving Requests ..... 145





18.2.2 Sending Responses ..... 146
18.3 Framing ..... 147
18.4 Error Handling ..... 147
19 Common Message Components ..... 147
19.1 SIP and SIPS Uniform Resource Indicators ..... 148
19.1.1 SIP and SIPS URI Components ..... 148
19.1.2 Character Escaping Requirements ..... 152
19.1.3 Example SIP and SIPS URIs ..... 153
19.1.4 URI Comparison ..... 153
19.1.5 Forming Requests from a URI ..... 156
19.1.6 Relating SIP URIs and tel URLs ..... 157
19.2 Option Tags ..... 158
19.3 Tags ..... 159
20 Header Fields ..... 159
20.1 Accept ..... 161
20.2 Accept-Encoding ..... 163
20.3 Accept-Language ..... 164
20.4 Alert-Info ..... 164
20.5 Allow ..... 165
20.6 Authentication-Info ..... 165

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 644
Keysight, EX-1045
Page 644

20.7 Authorization ..... 165
20.8 Call-ID ..... 166
20.9 Call-Info ..... 166
20.10 Contact ..... 167
20.11 Content-Disposition ..... 168
20.12 Content-Encoding ..... 169
20.13 Content-Language ..... 169
20.14 Content-Length ..... 169
20.15 Content-Type ..... 170
20.16 CSeq ..... 170
20.17 Date ..... 170
20.18 Error-Info ..... 171
20.19 Expires ..... 171
20.20 From ..... 172
20.21 In-Reply-To ..... 172
20.22 Max-Forwards ..... 173
20.23 Min-Expires ..... 173
20.24 MIME-Version ..... 173
20.25 Organization ..... 174
20.26 Priority ..... 174
20.27 Proxy-Authenticate ..... 174
20.28 Proxy-Authorization ..... 175
20.29 Proxy-Require ..... 175
20.30 Record-Route ..... 175
20.31 Reply-To ..... 176
20.32 Require ..... 176
20.33 Retry-After ..... 176
20.34 Route ..... 177





20.35 Server ..... 177
20.36 Subject ..... 177
20.37 Supported ..... 178
20.38 Timestamp ..... 178
20.39 To ..... 178
20.40 Unsupported ..... 179
20.41 User-Agent ..... 179
20.42 Via ..... 179
20.43 Warning ..... 180
20.44 WWW-Authenticate ..... 182
21 Response Codes ..... 182
21.1 Provisional 1xx ..... 182
21.1.1 100 Trying ..... 183
21.1.2 180 Ringing ..... 183
21.1.3 181 Call Is Being Forwarded ..... 183
21.1.4 182 Queued ..... 183
21.1.5 183 Session Progress ..... 183
21.2 Successful 2xx ..... 183
21.2.1 200 OK ..... 183
21.3 Redirection 3xx ..... 184
21.3.1 300 Multiple Choices ..... 184
21.3.2 301 Moved Permanently ..... 184
21.3.3 302 Moved Temporarily ..... 184
21.3.4 305 Use Proxy ..... 185
21.3.5 380 Alternative Service ..... 185
21.4 Request Failure 4xx ..... 185
21.4.1 400 Bad Request ..... 185

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 645
Keysight, EX-1045
Page 645

21.4.2 401 Unauthorized ..... 185
21.4.3 402 Payment Required ..... 186
21.4.4 403 Forbidden ..... 186
21.4.5 404 Not Found ..... 186
21.4.6 405 Method Not Allowed ..... 186
21.4.7 406 Not Acceptable ..... 186
21.4.8 407 Proxy Authentication Required ..... 186
21.4.9 408 Request Timeout ..... 186
21.4.10 410 Gone ..... 187
21.4.11 413 Request Entity Too Large ..... 187
21.4.12 414 Request-URI Too Long ..... 187
21.4.13 415 Unsupported Media Type ..... 187
21.4.14 416 Unsupported URI Scheme ..... 187
21.4.15 420 Bad Extension ..... 187
21.4.16 421 Extension Required ..... 188
21.4.17 423 Interval Too Brief ..... 188
21.4.18 480 Temporarily Unavailable ..... 188
21.4.19 481 Call/Transaction Does Not Exist ..... 188
21.4.20 482 Loop Detected ..... 188
21.4.21 483 Too Many Hops ..... 189
21.4.22 484 Address Incomplete ..... 189





21.4.23 485 Ambiguous ..... 189
21.4.24 486 Busy Here ..... 189
21.4.25 487 Request Terminated ..... 190
21.4.26 488 Not Acceptable Here ..... 190
21.4.27 491 Request Pending ..... 190
21.4.28 493 Undecipherable ..... 190
21.5 Server Failure 5xx ..... 190
21.5.1 500 Server Internal Error ..... 190
21.5.2 501 Not Implemented ..... 191
21.5.3 502 Bad Gateway ..... 191
21.5.4 503 Service Unavailable ..... 191
21.5.5 504 Server Time-out ..... 191
21.5.6 505 Version Not Supported ..... 192
21.5.7 513 Message Too Large ..... 192
21.6 Global Failures 6xx ..... 192
21.6.1 600 Busy Everywhere ..... 192
21.6.2 603 Decline ..... 192
21.6.3 604 Does Not Exist Anywhere ..... 192
21.6.4 606 Not Acceptable ..... 192
22 Usage of HTTP Authentication ..... 193
22.1 Framework ..... 193
22.2 User-to-User Authentication ..... 195
22.3 Proxy-to-User Authentication ..... 197
22.4 The Digest Authentication Scheme ..... 199
23 S/MIME ..... 201
23.1 S/MIME Certificates ..... 201
23.2 S/MIME Key Exchange ..... 202
23.3 Securing MIME bodies ..... 205
23.4 SIP Header Privacy and Integrity using S/MIME: Tunneling SIP ..... 207
23.4.1 Integrity and Confidentiality Properties of SIP Headers ..... 207
23.4.1.1 Integrity ..... 207
23.4.1.2 Confidentiality ..... 208

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 646
Keysight, EX-1045
Page 646

23.4.2 Tunneling Integrity and Authentication ..... 209
23.4.3 Tunneling Encryption ..... 211
24 Examples ..... 213
24.1 Registration ..... 213
24.2 Session Setup ..... 214
25 Augmented BNF for the SIP Protocol ..... 219
25.1 Basic Rules ..... 219
26 Security Considerations: Threat Model and Security Usage Recommendations ..... 232
26.1 Attacks and Threat Models ..... 233
26.1.1 Registration Hijacking ..... 233
26.1.2 Impersonating a Server ..... 234
26.1.3 Tampering with Message Bodies ..... 235
26.1.4 Tearing Down Sessions ..... 235





26.1.5 Denial of Service and Amplification ..... 236
26.2 Security Mechanisms ..... 237
26.2.1 Transport and Network Layer Security ..... 238
26.2.2 SIPS URI Scheme ..... 239
26.2.3 HTTP Authentication ..... 240
26.2.4 S/MIME ..... 240
26.3 Implementing Security Mechanisms ..... 241
26.3.1 Requirements for Implementers of SIP ..... 241
26.3.2 Security Solutions ..... 242
26.3.2.1 Registration ..... 242
26.3.2.2 Interdomain Requests ..... 243
26.3.2.3 Peer-to-Peer Requests ..... 245
26.3.2.4 DoS Protection ..... 246
26.4 Limitations ..... 247
26.4.1 HTTP Digest ..... 247
26.4.2 S/MIME ..... 248
26.4.3 TLS ..... 249
26.4.4 SIPS URIs ..... 249
26.5 Privacy ..... 251
27 IANA Considerations ..... 252
27.1 Option Tags ..... 252
27.2 Warn-Codes ..... 252
27.3 Header Field Names ..... 253
27.4 Method and Response Codes ..... 253
27.5 The "message/sip" MIME type. ..... 254
27.6 New Content-Disposition Parameter Registrations ..... 255
28 Changes From RFC 2543 ..... 255
28.1 Major Functional Changes ..... 255
28.2 Minor Functional Changes ..... 260
29 Normative References ..... 261
30 Informative References ..... 262
A Table of Timer Values ..... 265
Acknowledgments ..... 266
Authors' Addresses ..... 267
Full Copyright Statement ..... 269

## 1 Introduction

There are many applications of the Internet that require the creation and management of a session, where a session is considered an exchange of data between an association of participants. The

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 647
Keysight, EX-1045
Page 647

implementation of these applications is complicated by the practices of participants: users may move between endpoints, they may be addressable by multiple names, and they may communicate in several different media - sometimes simultaneously. Numerous protocols have been authored that carry various forms of real-time multimedia session data such as voice, video, or text messages. The Session Initiation Protocol (SIP) works in concert with these protocols by



enabling Internet endpoints (called user agents) to discover one another and to agree on a characterization of a session they would like to share. For locating prospective session participants, and for other functions, SIP enables the creation of an infrastructure of network hosts (called proxy servers) to which user agents can send registrations, invitations to sessions, and other requests. SIP is an agile, general-purpose tool for creating, modifying, and terminating sessions that works independently of underlying transport protocols and without dependency on the type of session that is being established.

## 2 Overview of SIP Functionality

SIP is an application-layer control protocol that can establish, modify, and terminate multimedia sessions (conferences) such as Internet telephony calls. SIP can also invite participants to already existing sessions, such as multicast conferences. Media can be added to (and removed from) an existing session. SIP transparently supports name mapping and redirection services, which supports personal mobility [27] - users can maintain a single externally visible identifier regardless of their network location.

SIP supports five facets of establishing and terminating multimedia communications:

- User location: determination of the end system to be used for communication;
- User availability: determination of the willingness of the called party to engage in communications;
- User capabilities: determination of the media and media parameters to be used;
- Session setup: "ringing", establishment of session parameters at both called and calling party;
- Session management: including transfer and termination of sessions, modifying session parameters, and invoking services.

SIP is not a vertically integrated communications system. SIP is rather a component that can be used with other IETF protocols to build a complete multimedia architecture. Typically, these architectures will include protocols such as the Real-time Transport Protocol (RTP) (RFC 1889 [28]) for transporting real-time data and providing QoS feedback, the Real-Time streaming protocol (RTSP) (RFC 2326 [29]) for controlling delivery of streaming media, the Media

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

Gateway Control Protocol (MEGACO) (RFC 3015 [30]) for controlling gateways to the Public Switched Telephone Network (PSTN), and the Session Description Protocol (SDP) (RFC 2327 [1]) for describing multimedia sessions. Therefore, SIP should be used in conjunction with other protocols in order to provide complete services to the users. However, the basic functionality and operation of SIP does not depend on any of these protocols.

SIP does not provide services. Rather, SIP provides primitives that can be used to implement different services. For example, SIP can locate a user and deliver an opaque object to his current location. If this primitive is used to deliver a session description written in SDP, for instance, the endpoints can agree on the parameters of a session. If the same primitive is used to deliver a photo of the caller as well as the session description, a "caller ID" service can be easily implemented. As this example shows, a single primitive is typically used to provide several different services.

SIP does not offer conference control services such as floor control or voting and does not prescribe how a conference is to be managed. SIP can be used to initiate a session that uses some other conference control protocol. Since SIP messages and the sessions they establish can pass through entirely different networks, SIP cannot, and does not, provide any kind of network resource reservation capabilities.

The nature of the services provided make security particularly important. To that end, SIP provides a suite of security services, which include denial-of-service prevention, authentication (both user to user and proxy to user), integrity protection, and encryption and privacy services.

SIP works with both IPv4 and IPv6.

## 3 Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [2] and indicate requirement levels for compliant SIP implementations.

## 4 Overview of Operation

This section introduces the basic operations of SIP using simple examples. This section is tutorial in nature and does not contain any normative statements.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 649
Keysight, EX-1045
Page 649

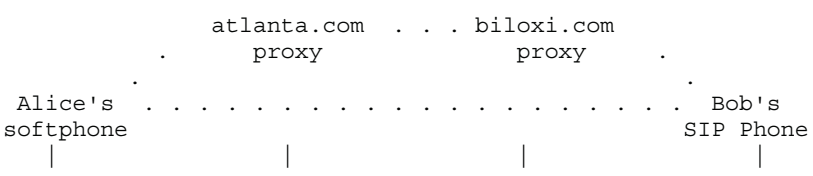
The first example shows the basic functions of SIP: location of an end point, signal of a desire to communicate, negotiation of session parameters to establish the session, and teardown of the session once established.

Figure 1 shows a typical example of a SIP message exchange between two users, Alice and Bob. (Each message is labeled with the letter "F" and a number for reference by the text.) In this example, Alice uses a SIP application on her PC (referred to as a softphone) to call Bob on his SIP phone over the Internet. Also shown are two SIP proxy servers that act on behalf of Alice and Bob to facilitate the session establishment. This typical arrangement is often referred to as the "SIP trapezoid" as shown by the geometric shape of the dotted lines in Figure 1.

Alice "calls" Bob using his SIP identity, a type of Uniform Resource Identifier (URI) called a SIP URI. SIP URIs are defined in Section 19.1. It has a similar form to an email address, typically containing a username and a host name. In this case, it is sip:bob@biloxi.com, where biloxi.com is the domain of Bob's SIP service provider. Alice has a SIP URI of sip:alice@atlanta.com. Alice might have typed in Bob's URI or perhaps clicked on a hyperlink or an entry in an address book. SIP also provides a secure URI, called a SIPS URI. An example would be sips:bob@biloxi.com. A call made to a SIPS URI guarantees that secure, encrypted transport (namely TLS) is used to carry all SIP messages from the caller to the domain of the callee. From there, the request is sent securely to the callee, but with security mechanisms that depend on the policy of the domain of the callee.

SIP is based on an HTTP-like request/response transaction model. Each transaction consists of a request that invokes a particular method, or function, on the server and at least one response. In this example, the transaction begins with Alice's softphone sending an INVITE request addressed to Bob's SIP URI. INVITE is an example of a SIP method that specifies the action that the requestor (Alice) wants the server (Bob) to take. The INVITE request contains a number of header fields. Header fields are named attributes that provide additional information about a message. The ones present in an INVITE include a unique identifier for the call, the destination address, Alice's address, and information about the type of session that Alice wishes to establish with Bob. The INVITE (message F1 in Figure 1) might look like this:






https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 650
Keysight, EX-1045
Page 650

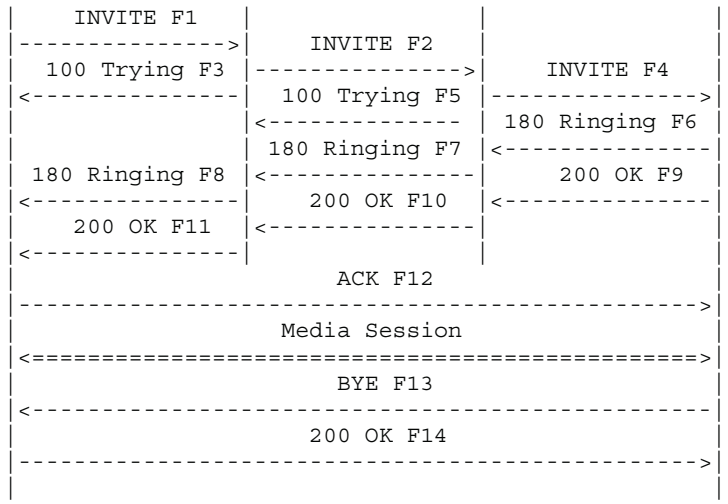



Figure 1: SIP session setup example with SIP trapezoid

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142

(Alice's SDP not shown)
```

The first line of the text-encoded message contains the method name (INVITE). The lines that follow are a list of header fields. This example contains a minimum required set. The header fields are briefly described below:





Via contains the address (pc33.atlanta.com) at which Alice is expecting to receive responses to this request. It also contains a branch parameter that identifies this transaction.

To contains a display name (Bob) and a SIP or SIPS URI (sip:bob@biloxi.com) towards which the request was originally directed. Display names are described in RFC 2822 [3].

From also contains a display name (Alice) and a SIP or SIPS URI (sip:alice@atlanta.com) that indicate the originator of the request. This header field also has a tag parameter containing a random string (1928301774) that was added to the URI by the softphone. It is used for identification purposes.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 651
Keysight, EX-1045
Page 651

Call-ID contains a globally unique identifier for this call, generated by the combination of a random string and the softphone's host name or IP address. The combination of the To tag, From tag, and Call-ID completely defines a peer-to-peer SIP relationship between Alice and Bob and is referred to as a dialog.

CSeq or Command Sequence contains an integer and a method name. The CSeq number is incremented for each new request within a dialog and is a traditional sequence number.

Contact contains a SIP or SIPS URI that represents a direct route to contact Alice, usually composed of a username at a fully qualified domain name (FQDN). While an FQDN is preferred, many end systems do not have registered domain names, so IP addresses are permitted. While the Via header field tells other elements where to send the response, the Contact header field tells other elements where to send future requests.

Max-Forwards serves to limit the number of hops a request can make on the way to its destination. It consists of an integer that is decremented by one at each hop.

Content-Type contains a description of the message body (not shown).

Content-Length contains an octet (byte) count of the message body.

The complete set of SIP header fields is defined in Section 20.

The details of the session, such as the type of media, codec, or sampling rate, are not described using SIP. Rather, the body of a SIP message contains a description of the session, encoded in some other protocol format. One such format is the Session Description Protocol (SDP) (RFC 2327 [1]). This SDP message (not shown in the





example) is carried by the SIP message in a way that is analogous to a document attachment being carried by an email message, or a web page being carried in an HTTP message.

Since the softphone does not know the location of Bob or the SIP server in the biloxi.com domain, the softphone sends the INVITE to the SIP server that serves Alice's domain, atlanta.com. The address of the atlanta.com SIP server could have been configured in Alice's softphone, or it could have been discovered by DHCP, for example.

The atlanta.com SIP server is a type of SIP server known as a proxy server. A proxy server receives SIP requests and forwards them on behalf of the requestor. In this example, the proxy server receives the INVITE request and sends a 100 (Trying) response back to Alice's softphone. The 100 (Trying) response indicates that the INVITE has been received and that the proxy is working on her behalf to route the INVITE to the destination. Responses in SIP use a three-digit code followed by a descriptive phrase. This response contains the same To, From, Call-ID, CSeq and branch parameter in the Via as the INVITE, which allows Alice's softphone to correlate this response to

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

the sent INVITE. The atlanta.com proxy server locates the proxy server at biloxi.com, possibly by performing a particular type of DNS (Domain Name Service) lookup to find the SIP server that serves the biloxi.com domain. This is described in [4]. As a result, it obtains the IP address of the biloxi.com proxy server and forwards, or proxies, the INVITE request there. Before forwarding the request, the atlanta.com proxy server adds an additional Via header field value that contains its own address (the INVITE already contains Alice's address in the first Via). The biloxi.com proxy server receives the INVITE and responds with a 100 (Trying) response back to the atlanta.com proxy server to indicate that it has received the INVITE and is processing the request. The proxy server consults a database, generically called a location service, that contains the current IP address of Bob. (We shall see in the next section how this database can be populated.) The biloxi.com proxy server adds another Via header field value with its own address to the INVITE and proxies it to Bob's SIP phone.

Bob's SIP phone receives the INVITE and alerts Bob to the incoming call from Alice so that Bob can decide whether to answer the call, that is, Bob's phone rings. Bob's SIP phone indicates this in a 180 (Ringing) response, which is routed back through the two proxies in the reverse direction. Each proxy uses the Via header field to determine where to send the response and removes its own address from the top. As a result, although DNS and location service lookups were required to route the initial INVITE, the 180 (Ringing) response can be returned to the caller without lookups or without state being





maintained in the proxies. This also has the desirable property that each proxy that sees the INVITE will also see all responses to the INVITE.

When Alice's softphone receives the 180 (Ringing) response, it passes this information to Alice, perhaps using an audio ringback tone or by displaying a message on Alice's screen.

In this example, Bob decides to answer the call. When he picks up the handset, his SIP phone sends a 200 (OK) response to indicate that the call has been answered. The 200 (OK) contains a message body with the SDP media description of the type of session that Bob is willing to establish with Alice. As a result, there is a two-phase exchange of SDP messages: Alice sent one to Bob, and Bob sent one back to Alice. This two-phase exchange provides basic negotiation capabilities and is based on a simple offer/answer model of SDP exchange. If Bob did not wish to answer the call or was busy on another call, an error response would have been sent instead of the 200 (OK), which would have resulted in no media session being established. The complete list of SIP response codes is in Section 21. The 200 (OK) (message F9 in Figure 1) might look like this as Bob sends it out:

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP server10.biloxi.com
   ;branch=z9hG4bKnashds8;received=192.0.2.3
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

```
      ;branch=z9hG4bK77ef4c2312983.1;received=192.0.2.2
   Via: SIP/2.0/UDP pc33.atlanta.com
      ;branch=z9hG4bK776asdhds ;received=192.0.2.1
   To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
   From: Alice <sip:alice@atlanta.com>;tag=1928301774
   Call-ID: a84b4c76e66710@pc33.atlanta.com
   CSeq: 314159 INVITE
   Contact: <sip:bob@192.0.2.4>
   Content-Type: application/sdp
   Content-Length: 131

   (Bob's SDP not shown)
```

The first line of the response contains the response code (200) and the reason phrase (OK). The remaining lines contain header fields. The Via, To, From, Call-ID, and CSeq header fields are copied from the INVITE request. (There are three Via header field values - one added by Alice's SIP phone, one added by the atlanta.com proxy, and one added by the biloxi.com proxy.) Bob's SIP phone has added a tag parameter to the To header field. This tag will be incorporated by both endpoints into the dialog and will be included in all future





requests and responses in this call. The Contact header field contains a URI at which Bob can be directly reached at his SIP phone. The Content-Type and Content-Length refer to the message body (not shown) that contains Bob's SDP media information.

In addition to DNS and location service lookups shown in this example, proxy servers can make flexible "routing decisions" to decide where to send a request. For example, if Bob's SIP phone returned a 486 (Busy Here) response, the biloxi.com proxy server could proxy the INVITE to Bob's voicemail server. A proxy server can also send an INVITE to a number of locations at the same time. This type of parallel search is known as forking.

In this case, the 200 (OK) is routed back through the two proxies and is received by Alice's softphone, which then stops the ringback tone and indicates that the call has been answered. Finally, Alice's softphone sends an acknowledgement message, ACK, to Bob's SIP phone to confirm the reception of the final response (200 (OK)). In this example, the ACK is sent directly from Alice's softphone to Bob's SIP phone, bypassing the two proxies. This occurs because the endpoints have learned each other's address from the Contact header fields through the INVITE/200 (OK) exchange, which was not known when the initial INVITE was sent. The lookups performed by the two proxies are no longer needed, so the proxies drop out of the call flow. This completes the INVITE/200/ACK three-way handshake used to establish SIP sessions. Full details on session setup are in Section 13.

Alice and Bob's media session has now begun, and they send media packets using the format to which they agreed in the exchange of SDP. In general, the end-to-end media packets take a different path from the SIP signaling messages.

During the session, either Alice or Bob may decide to change the characteristics of the media session. This is accomplished by

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 654
Keysight, EX-1045
Page 654

sending a re-INVITE containing a new media description. This re-INVITE references the existing dialog so that the other party knows that it is to modify an existing session instead of establishing a new session. The other party sends a 200 (OK) to accept the change. The requestor responds to the 200 (OK) with an ACK. If the other party does not accept the change, he sends an error response such as 488 (Not Acceptable Here), which also receives an ACK. However, the failure of the re-INVITE does not cause the existing call to fail - the session continues using the previously negotiated characteristics. Full details on session modification are in Section 14.





At the end of the call, Bob disconnects (hangs up) first and generates a BYE message. This BYE is routed directly to Alice's softphone, again bypassing the proxies. Alice confirms receipt of the BYE with a 200 (OK) response, which terminates the session and the BYE transaction. No ACK is sent - an ACK is only sent in response to a response to an INVITE request. The reasons for this special handling for INVITE will be discussed later, but relate to the reliability mechanisms in SIP, the length of time it can take for a ringing phone to be answered, and forking. For this reason, request handling in SIP is often classified as either INVITE or non-INVITE, referring to all other methods besides INVITE. Full details on session termination are in Section 15.

Section 24.2 describes the messages shown in Figure 1 in full.

In some cases, it may be useful for proxies in the SIP signaling path to see all the messaging between the endpoints for the duration of the session. For example, if the biloxi.com proxy server wished to remain in the SIP messaging path beyond the initial INVITE, it would add to the INVITE a required routing header field known as Record-Route that contained a URI resolving to the hostname or IP address of the proxy. This information would be received by both Bob's SIP phone and (due to the Record-Route header field being passed back in the 200 (OK)) Alice's softphone and stored for the duration of the dialog. The biloxi.com proxy server would then receive and proxy the ACK, BYE, and 200 (OK) to the BYE. Each proxy can independently decide to receive subsequent messages, and those messages will pass through all proxies that elect to receive it. This capability is frequently used for proxies that are providing mid-call features.

Registration is another common operation in SIP. Registration is one way that the biloxi.com server can learn the current location of Bob. Upon initialization, and at periodic intervals, Bob's SIP phone sends REGISTER messages to a server in the biloxi.com domain known as a SIP registrar. The REGISTER messages associate Bob's SIP or SIPS URI (sip:bob@biloxi.com) with the machine into which he is currently logged (conveyed as a SIP or SIPS URI in the Contact header field). The registrar writes this association, also called a binding, to a database, called the location service, where it can be used by the proxy in the biloxi.com domain. Often, a registrar server for a domain is co-located with the proxy for that domain. It is an

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 655
Keysight, EX-1045
Page 655

important concept that the distinction between types of SIP servers is logical, not physical.

Bob is not limited to registering from a single device. For example, both his SIP phone at home and the one in the office could send registrations. This information is stored together in the location





service and allows a proxy to perform various types of searches to locate Bob. Similarly, more than one user can be registered on a single device at the same time.

The location service is just an abstract concept. It generally contains information that allows a proxy to input a URI and receive a set of zero or more URIs that tell the proxy where to send the request. Registrations are one way to create this information, but not the only way. Arbitrary mapping functions can be configured at the discretion of the administrator.

Finally, it is important to note that in SIP, registration is used for routing incoming SIP requests and has no role in authorizing outgoing requests. Authorization and authentication are handled in SIP either on a request-by-request basis with a challenge/response mechanism, or by using a lower layer scheme as discussed in Section 26.

The complete set of SIP message details for this registration example is in Section 24.1.

Additional operations in SIP, such as querying for the capabilities of a SIP server or client using OPTIONS, or canceling a pending request using CANCEL, will be introduced in later sections.

## 5 Structure of the Protocol

SIP is structured as a layered protocol, which means that its behavior is described in terms of a set of fairly independent processing stages with only a loose coupling between each stage. The protocol behavior is described as layers for the purpose of presentation, allowing the description of functions common across elements in a single section. It does not dictate an implementation in any way. When we say that an element "contains" a layer, we mean it is compliant to the set of rules defined by that layer.

Not every element specified by the protocol contains every layer. Furthermore, the elements specified by SIP are logical elements, not physical ones. A physical realization can choose to act as different logical elements, perhaps even on a transaction-by-transaction basis.

The lowest layer of SIP is its syntax and encoding. Its encoding is specified using an augmented Backus-Naur Form grammar (BNF). The complete BNF is specified in Section 25; an overview of a SIP message's structure can be found in Section 7.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 656
Keysight, EX-1045
Page 656

The second layer is the transport layer. It defines how a client sends requests and receives responses and how a server receives requests and sends responses over the network. All SIP elements contain a transport layer. The transport layer is described in Section 18.

The third layer is the transaction layer. Transactions are a fundamental component of SIP. A transaction is a request sent by a client transaction (using the transport layer) to a server transaction, along with all responses to that request sent from the server transaction back to the client. The transaction layer handles application-layer retransmissions, matching of responses to requests, and application-layer timeouts. Any task that a user agent client (UAC) accomplishes takes place using a series of transactions. Discussion of transactions can be found in Section 17. User agents contain a transaction layer, as do stateful proxies. Stateless proxies do not contain a transaction layer. The transaction layer has a client component (referred to as a client transaction) and a server component (referred to as a server transaction), each of which are represented by a finite state machine that is constructed to process a particular request.

The layer above the transaction layer is called the transaction user (TU). Each of the SIP entities, except the stateless proxy, is a transaction user. When a TU wishes to send a request, it creates a client transaction instance and passes it the request along with the destination IP address, port, and transport to which to send the request. A TU that creates a client transaction can also cancel it. When a client cancels a transaction, it requests that the server stop further processing, revert to the state that existed before the transaction was initiated, and generate a specific error response to that transaction. This is done with a CANCEL request, which constitutes its own transaction, but references the transaction to be cancelled (Section 9).

The SIP elements, that is, user agent clients and servers, stateless and stateful proxies and registrars, contain a core that distinguishes them from each other. Cores, except for the stateless proxy, are transaction users. While the behavior of the UAC and UAS cores depends on the method, there are some common rules for all methods (Section 8). For a UAC, these rules govern the construction of a request; for a UAS, they govern the processing of a request and generating a response. Since registrations play an important role in SIP, a UAS that handles a REGISTER is given the special name registrar. Section 10 describes UAC and UAS core behavior for the REGISTER method. Section 11 describes UAC and UAS core behavior for the OPTIONS method, used for determining the capabilities of a UA.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 657
Keysight, EX-1045
Page 657

Certain other requests are sent within a dialog. A dialog is a peer-to-peer SIP relationship between two user agents that persists for some time. The dialog facilitates sequencing of messages and proper routing of requests between the user agents. The INVITE method is the only way defined in this specification to establish a dialog. When a UAC sends a request that is within the context of a dialog, it follows the common UAC rules as discussed in Section 8 but also the rules for mid-dialog requests. Section 12 discusses dialogs and presents the procedures for their construction and maintenance, in addition to construction of requests within a dialog.

The most important method in SIP is the INVITE method, which is used to establish a session between participants. A session is a collection of participants, and streams of media between them, for the purposes of communication. Section 13 discusses how sessions are initiated, resulting in one or more SIP dialogs. Section 14 discusses how characteristics of that session are modified through the use of an INVITE request within a dialog. Finally, section 15 discusses how a session is terminated.

The procedures of Sections 8, 10, 11, 12, 13, 14, and 15 deal entirely with the UA core (Section 9 describes cancellation, which applies to both UA core and proxy core). Section 16 discusses the proxy element, which facilitates routing of messages between user agents.

## 6 Definitions

The following terms have special significance for SIP.

Address-of-Record: An address-of-record (AOR) is a SIP or SIPS URI that points to a domain with a location service that can map the URI to another URI where the user might be available. Typically, the location service is populated through registrations. An AOR is frequently thought of as the "public address" of the user.

Back-to-Back User Agent: A back-to-back user agent (B2BUA) is a logical entity that receives a request and processes it as a user agent server (UAS). In order to determine how the request should be answered, it acts as a user agent client (UAC) and generates requests. Unlike a proxy server, it maintains dialog state and must participate in all requests sent on the dialogs it has established. Since it is a concatenation of a UAC and UAS, no explicit definitions are needed for its behavior.



Call: A call is an informal term that refers to some communication between peers, generally set up for the purposes of a multimedia conversation.

Call Leg: Another name for a dialog [31]; no longer used in this specification.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 658
Keysight, EX-1045
Page 658

Call Stateful: A proxy is call stateful if it retains state for a dialog from the initiating INVITE to the terminating BYE request. A call stateful proxy is always transaction stateful, but the converse is not necessarily true.

Client: A client is any network element that sends SIP requests and receives SIP responses. Clients may or may not interact directly with a human user. User agent clients and proxies are clients.

Conference: A multimedia session (see below) that contains multiple participants.

Core: Core designates the functions specific to a particular type of SIP entity, i.e., specific to either a stateful or stateless proxy, a user agent or registrar. All cores, except those for the stateless proxy, are transaction users.

Dialog: A dialog is a peer-to-peer SIP relationship between two UAs that persists for some time. A dialog is established by SIP messages, such as a 2xx response to an INVITE request. A dialog is identified by a call identifier, local tag, and a remote tag. A dialog was formerly known as a call leg in RFC 2543.

Downstream: A direction of message forwarding within a transaction that refers to the direction that requests flow from the user agent client to user agent server.

Final Response: A response that terminates a SIP transaction, as opposed to a provisional response that does not. All 2xx, 3xx, 4xx, 5xx and 6xx responses are final.

Header: A header is a component of a SIP message that conveys information about the message. It is structured as a sequence of header fields.

Header Field: A header field is a component of the SIP message header. A header field can appear as one or more header field rows. Header field rows consist of a header field name and zero or more header field values. Multiple header field values on a





given header field row are separated by commas. Some header fields can only have a single header field value, and as a result, always appear as a single header field row.

Header Field Value: A header field value is a single value; a header field consists of zero or more header field values.

Home Domain: The domain providing service to a SIP user. Typically, this is the domain present in the URI in the address-of-record of a registration.

Informational Response: Same as a provisional response.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

Initiator, Calling Party, Caller: The party initiating a session (and dialog) with an INVITE request. A caller retains this role from the time it sends the initial INVITE that established a dialog until the termination of that dialog.

Invitation: An INVITE request.

Invitee, Invited User, Called Party, Callee: The party that receives an INVITE request for the purpose of establishing a new session. A callee retains this role from the time it receives the INVITE until the termination of the dialog established by that INVITE.

Location Service: A location service is used by a SIP redirect or proxy server to obtain information about a callee's possible location(s). It contains a list of bindings of address-of-record keys to zero or more contact addresses. The bindings can be created and removed in many ways; this specification defines a REGISTER method that updates the bindings.

Loop: A request that arrives at a proxy, is forwarded, and later arrives back at the same proxy. When it arrives the second time, its Request-URI is identical to the first time, and other header fields that affect proxy operation are unchanged, so that the proxy would make the same processing decision on the request it made the first time. Looped requests are errors, and the procedures for detecting them and handling them are described by the protocol.

Loose Routing: A proxy is said to be loose routing if it follows the procedures defined in this specification for processing of the Route header field. These procedures separate the destination of the request (present in the Request-URI) from





the set of proxies that need to be visited along the way (present in the Route header field). A proxy compliant to these mechanisms is also known as a loose router.

Message: Data sent between SIP elements as part of the protocol. SIP messages are either requests or responses.

Method: The method is the primary function that a request is meant to invoke on a server. The method is carried in the request message itself. Example methods are INVITE and BYE.

Outbound Proxy: A proxy that receives requests from a client, even though it may not be the server resolved by the Request-URI. Typically, a UA is manually configured with an outbound proxy, or can learn about one through auto-configuration protocols.

Parallel Search: In a parallel search, a proxy issues several requests to possible user locations upon receiving an incoming request. Rather than issuing one request and then waiting for the final response before issuing the next request as in a

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 660
Keysight, EX-1045
Page 660

sequential search, a parallel search issues requests without waiting for the result of previous requests.

Provisional Response: A response used by the server to indicate progress, but that does not terminate a SIP transaction. 1xx responses are provisional, other responses are considered final.

Proxy, Proxy Server: An intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients. A proxy server primarily plays the role of routing, which means its job is to ensure that a request is sent to another entity "closer" to the targeted user. Proxies are also useful for enforcing policy (for example, making sure a user is allowed to make a call). A proxy interprets, and, if necessary, rewrites specific parts of a request message before forwarding it.

Recursion: A client recurses on a 3xx response when it generates a new request to one or more of the URIs in the Contact header field in the response.

Redirect Server: A redirect server is a user agent server that generates 3xx responses to requests it receives, directing the client to contact an alternate set of URIs.





Registrar: A registrar is a server that accepts REGISTER requests and places the information it receives in those requests into the location service for the domain it handles.

Regular Transaction: A regular transaction is any transaction with a method other than INVITE, ACK, or CANCEL.

Request: A SIP message sent from a client to a server, for the purpose of invoking a particular operation.

Response: A SIP message sent from a server to a client, for indicating the status of a request sent from the client to the server.

Ringback: Ringback is the signaling tone produced by the calling party's application indicating that a called party is being alerted (ringing).

Route Set: A route set is a collection of ordered SIP or SIPS URI which represent a list of proxies that must be traversed when sending a particular request. A route set can be learned, through headers like Record-Route, or it can be configured.

Server: A server is a network element that receives requests in order to service them and sends back responses to those requests. Examples of servers are proxies, user agent servers, redirect servers, and registrars.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 661
Keysight, EX-1045
Page 661

Sequential Search: In a sequential search, a proxy server attempts each contact address in sequence, proceeding to the next one only after the previous has generated a final response. A 2xx or 6xx class final response always terminates a sequential search.

Session: From the SDP specification: "A multimedia session is a set of multimedia senders and receivers and the data streams flowing from senders to receivers. A multimedia conference is an example of a multimedia session." (RFC 2327 [1]) (A session as defined for SDP can comprise one or more RTP sessions.) As defined, a callee can be invited several times, by different calls, to the same session. If SDP is used, a session is defined by the concatenation of the SDP user name, session id, network type, address type, and address elements in the origin field.

SIP Transaction: A SIP transaction occurs between a client and a server and comprises all messages from the first request sent from the client to the server up to a final (non-1xx) response





sent from the server to the client. If the request is INVITE and the final response is a non-2xx, the transaction also includes an ACK to the response. The ACK for a 2xx response to an INVITE request is a separate transaction.

Spiral: A spiral is a SIP request that is routed to a proxy, forwarded onwards, and arrives once again at that proxy, but this time differs in a way that will result in a different processing decision than the original request. Typically, this means that the request's Request-URI differs from its previous arrival. A spiral is not an error condition, unlike a loop. A typical cause for this is call forwarding. A user calls joe@example.com. The example.com proxy forwards it to Joe's PC, which in turn, forwards it to bob@example.com. This request is proxied back to the example.com proxy. However, this is not a loop. Since the request is targeted at a different user, it is considered a spiral, and is a valid condition.

Stateful Proxy: A logical entity that maintains the client and server transaction state machines defined by this specification during the processing of a request, also known as a transaction stateful proxy. The behavior of a stateful proxy is further defined in Section 16. A (transaction) stateful proxy is not the same as a call stateful proxy.

Stateless Proxy: A logical entity that does not maintain the client or server transaction state machines defined in this specification when it processes requests. A stateless proxy forwards every request it receives downstream and every response it receives upstream.

Strict Routing: A proxy is said to be strict routing if it follows the Route processing rules of RFC 2543 and many prior work in

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 662
Keysight, EX-1045
Page 662

progress versions of this RFC. That rule caused proxies to destroy the contents of the Request-URI when a Route header field was present. Strict routing behavior is not used in this specification, in favor of a loose routing behavior. Proxies that perform strict routing are also known as strict routers.

Target Refresh Request: A target refresh request sent within a dialog is defined as a request that can modify the remote target of the dialog.

Transaction User (TU): The layer of protocol processing that resides above the transaction layer. Transaction users include the UAC core, UAS core, and proxy core.





Upstream: A direction of message forwarding within a transaction that refers to the direction that responses flow from the user agent server back to the user agent client.

URL-encoded: A character string encoded according to RFC 2396, Section 2.4 [5].

User Agent Client (UAC): A user agent client is a logical entity that creates a new request, and then uses the client transaction state machinery to send it. The role of UAC lasts only for the duration of that transaction. In other words, if a piece of software initiates a request, it acts as a UAC for the duration of that transaction. If it receives a request later, it assumes the role of a user agent server for the processing of that transaction.

UAC Core: The set of processing functions required of a UAC that reside above the transaction and transport layers.

User Agent Server (UAS): A user agent server is a logical entity that generates a response to a SIP request. The response accepts, rejects, or redirects the request. This role lasts only for the duration of that transaction. In other words, if a piece of software responds to a request, it acts as a UAS for the duration of that transaction. If it generates a request later, it assumes the role of a user agent client for the processing of that transaction.

UAS Core: The set of processing functions required at a UAS that resides above the transaction and transport layers.

User Agent (UA): A logical entity that can act as both a user agent client and user agent server.

The role of UAC and UAS, as well as proxy and redirect servers, are defined on a transaction-by-transaction basis. For example, the user agent initiating a call acts as a UAC when sending the initial INVITE request and as a UAS when receiving a BYE request from the callee. Similarly, the same software can act as a proxy server for one request and as a redirect server for the next request.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 663
Keysight, EX-1045
Page 663

Proxy, location, and registrar servers defined above are logical entities; implementations MAY combine them into a single application.

## 7 SIP Messages

SIP is a text-based protocol and uses the UTF-8 charset (RFC 2279 [7]).



A SIP message is either a request from a client to a server, or a response from a server to a client.

Both Request (section 7.1) and Response (section 7.2) messages use the basic format of RFC 2822 [3], even though the syntax differs in character set and syntax specifics. (SIP allows header fields that would not be valid RFC 2822 header fields, for example.) Both types of messages consist of a start-line, one or more header fields, an empty line indicating the end of the header fields, and an optional message-body.

```
generic-message  =  start-line
                    *message-header
                    CRLF
                    [ message-body ]
start-line       =  Request-Line / Status-Line
```

The start-line, each message-header line, and the empty line MUST be terminated by a carriage-return line-feed sequence (CRLF). Note that the empty line MUST be present even if the message-body is not.

Except for the above difference in character sets, much of SIP's message and header field syntax is identical to HTTP/1.1. Rather than repeating the syntax and semantics here, we use [HX.Y] to refer to Section X.Y of the current HTTP/1.1 specification (RFC 2616 [8]).

However, SIP is not an extension of HTTP.

### 7.1 Requests

SIP requests are distinguished by having a Request-Line for a start-line. A Request-Line contains a method name, a Request-URI, and the protocol version separated by a single space (SP) character.

The Request-Line ends with CRLF. No CR or LF are allowed except in the end-of-line CRLF sequence. No linear whitespace (LWS) is allowed in any of the elements.

```
Request-Line  =  Method SP Request-URI SP SIP-Version CRLF
```

Method: This specification defines six methods: REGISTER for registering contact information, INVITE, ACK, and CANCEL for setting up sessions, BYE for terminating sessions, and OPTIONS for querying servers about their capabilities. SIP extensions, documented in standards track RFCs, may define additional methods.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

Request-URI: The Request-URI is a SIP or SIPS URI as described in Section 19.1 or a general URI (RFC 2396 [5]). It indicates the user or service to which this request is being addressed. The Request-URI MUST NOT contain unescaped spaces or control characters and MUST NOT be enclosed in "<>".

SIP elements MAY support Request-URIs with schemes other than "sip" and "sips", for example the "tel" URI scheme of RFC 2806 [9]. SIP elements MAY translate non-SIP URIs using any mechanism at their disposal, resulting in SIP URI, SIPS URI, or some other scheme.

SIP-Version: Both request and response messages include the version of SIP in use, and follow [H3.1] (with HTTP replaced by SIP, and HTTP/1.1 replaced by SIP/2.0) regarding version ordering, compliance requirements, and upgrading of version numbers. To be compliant with this specification, applications sending SIP messages MUST include a SIP-Version of "SIP/2.0". The SIP-Version string is case-insensitive, but implementations MUST send upper-case.

Unlike HTTP/1.1, SIP treats the version number as a literal string. In practice, this should make no difference.

## 7.2 Responses

SIP responses are distinguished from requests by having a Status-Line as their start-line. A Status-Line consists of the protocol version followed by a numeric Status-Code and its associated textual phrase, with each element separated by a single SP character.

No CR or LF is allowed except in the final CRLF sequence.

```
Status-Line  =  SIP-Version SP Status-Code SP Reason-Phrase CRLF
```

The Status-Code is a 3-digit integer result code that indicates the outcome of an attempt to understand and satisfy a request. The Reason-Phrase is intended to give a short textual description of the Status-Code. The Status-Code is intended for use by automata, whereas the Reason-Phrase is intended for the human user. A client is not required to examine or display the Reason-Phrase.

While this specification suggests specific wording for the reason phrase, implementations MAY choose other text, for example, in the language indicated in the Accept-Language header field of the request.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 665
Keysight, EX-1045
Page 665

The first digit of the Status-Code defines the class of response. The last two digits do not have any categorization role. For this reason, any response with a status code between 100 and 199 is referred to as a "1xx response", any response with a status code between 200 and 299 as a "2xx response", and so on. SIP/2.0 allows six values for the first digit:

1xx: Provisional -- request received, continuing to process the request;

2xx: Success -- the action was successfully received, understood, and accepted;

3xx: Redirection -- further action needs to be taken in order to complete the request;

4xx: Client Error -- the request contains bad syntax or cannot be fulfilled at this server;

5xx: Server Error -- the server failed to fulfill an apparently valid request;

6xx: Global Failure -- the request cannot be fulfilled at any server.

Section 21 defines these classes and describes the individual codes.

## 7.3 Header Fields

SIP header fields are similar to HTTP header fields in both syntax and semantics. In particular, SIP header fields follow the [H4.2] definitions of syntax for the message-header and the rules for extending header fields over multiple lines. However, the latter is specified in HTTP with implicit whitespace and folding. This specification conforms to RFC 2234 [10] and uses only explicit whitespace and folding as an integral part of the grammar.

[H4.2] also specifies that multiple header fields of the same field name whose value is a comma-separated list can be combined into one header field. That applies to SIP as well, but the specific rule is different because of the different grammars. Specifically, any SIP header whose grammar is of the form

```
header  =  "header-name" HCOLON header-value *(COMMA header-value)
```

allows for combining header fields of the same name into a comma-separated list. The Contact header field allows a comma-separated list unless the header field value is "*".



### 7.3.1 Header Field Format

Header fields follow the same generic header format as that given in Section 2.2 of RFC 2822 [3]. Each header field consists of a field name followed by a colon (":") and the field value.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 666
Keysight, EX-1045
Page 666

```
   field-name: field-value
```

The formal grammar for a message-header specified in Section 25 allows for an arbitrary amount of whitespace on either side of the colon; however, implementations should avoid spaces between the field name and the colon and use a single space (SP) between the colon and the field-value.

```
   Subject:            lunch
   Subject      :      lunch
   Subject            :lunch
   Subject: lunch
```

Thus, the above are all valid and equivalent, but the last is the preferred form.

Header fields can be extended over multiple lines by preceding each extra line with at least one SP or horizontal tab (HT). The line break and the whitespace at the beginning of the next line are treated as a single SP character. Thus, the following are equivalent:

```
   Subject: I know you're there, pick up the phone and talk to me!
   Subject: I know you're there,
            pick up the phone
            and talk to me!
```

The relative order of header fields with different field names is not significant. However, it is RECOMMENDED that header fields which are needed for proxy processing (Via, Route, Record-Route, Proxy-Require, Max-Forwards, and Proxy-Authorization, for example) appear towards the top of the message to facilitate rapid parsing. The relative order of header field rows with the same field name is important. Multiple header field rows with the same field-name MAY be present in a message if and only if the entire field-value for that header field is defined as a comma-separated list (that is, if follows the grammar defined in Section 7.3). It MUST be possible to combine the multiple header field rows into one "field-name: field-value" pair, without changing the semantics of the message, by appending each subsequent field-value to the first, each separated by a comma. The exceptions to this rule are the WWW-Authenticate, Authorization, Proxy-Authenticate, and Proxy-Authorization header fields. Multiple header





field rows with these names MAY be present in a message, but since their grammar does not follow the general form listed in Section 7.3, they MUST NOT be combined into a single header field row.

Implementations MUST be able to process multiple header field rows with the same name in any combination of the single-value-per-line or comma-separated value forms.

The following groups of header field rows are valid and equivalent:

```
   Route: <sip:alice@atlanta.com>
   Subject: Lunch
   Route: <sip:bob@biloxi.com>
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 667
Keysight, EX-1045
Page 667

```
   Route: <sip:carol@chicago.com>

   Route: <sip:alice@atlanta.com>, <sip:bob@biloxi.com>
   Route: <sip:carol@chicago.com>
   Subject: Lunch

   Subject: Lunch
   Route: <sip:alice@atlanta.com>, <sip:bob@biloxi.com>,
          <sip:carol@chicago.com>
```

Each of the following blocks is valid but not equivalent to the others:

```
   Route: <sip:alice@atlanta.com>
   Route: <sip:bob@biloxi.com>
   Route: <sip:carol@chicago.com>

   Route: <sip:bob@biloxi.com>
   Route: <sip:alice@atlanta.com>
   Route: <sip:carol@chicago.com>

   Route: <sip:alice@atlanta.com>,<sip:carol@chicago.com>,
          <sip:bob@biloxi.com>
```

The format of a header field-value is defined per header-name. It will always be either an opaque sequence of TEXT-UTF8 octets, or a combination of whitespace, tokens, separators, and quoted strings. Many existing header fields will adhere to the general form of a value followed by a semi-colon separated sequence of parameter-name, parameter-value pairs:

```
      field-name: field-value *(;parameter-name=parameter-value)
```





Even though an arbitrary number of parameter pairs may be attached to a header field value, any given parameter-name MUST NOT appear more than once.

When comparing header fields, field names are always case-insensitive. Unless otherwise stated in the definition of a particular header field, field values, parameter names, and parameter values are case-insensitive. Tokens are always case-insensitive. Unless specified otherwise, values expressed as quoted strings are case-sensitive. For example,

```
   Contact: <sip:alice@atlanta.com>;expires=3600
```

is equivalent to

```
   CONTACT: <sip:alice@atlanta.com>;ExPiReS=3600
```

and

```
   Content-Disposition: session;handling=optional
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 668
Keysight, EX-1045
Page 668

is equivalent to

```
content-disposition: Session;HANDLING=OPTIONAL
```

The following two header fields are not equivalent:

```
Warning: 370 devnull "Choose a bigger pipe"
Warning: 370 devnull "CHOOSE A BIGGER PIPE"
```

#### 7.3.2 Header Field Classification

Some header fields only make sense in requests or responses. These are called request header fields and response header fields, respectively. If a header field appears in a message not matching its category (such as a request header field in a response), it MUST be ignored. Section 20 defines the classification of each header field.

#### 7.3.3 Compact Form

SIP provides a mechanism to represent common header field names in an abbreviated form. This may be useful when messages would otherwise become too large to be carried on the transport available to it (exceeding the maximum transmission unit (MTU) when using UDP, for example). These compact forms are defined in Section 20. A compact form MAY be substituted for the longer form of a header field name at any time without changing the semantics of the message. A header





field name MAY appear in both long and short forms within the same message. Implementations MUST accept both the long and short forms of each header name.

### 7.4 Bodies

Requests, including new requests defined in extensions to this specification, MAY contain message bodies unless otherwise noted. The interpretation of the body depends on the request method.

For response messages, the request method and the response status code determine the type and interpretation of any message body. All responses MAY include a body.

#### 7.4.1 Message Body Type

The Internet media type of the message body MUST be given by the Content-Type header field. If the body has undergone any encoding such as compression, then this MUST be indicated by the Content-Encoding header field; otherwise, Content-Encoding MUST be omitted. If applicable, the character set of the message body is indicated as part of the Content-Type header-field value.

The "multipart" MIME type defined in RFC 2046 [11] MAY be used within the body of the message. Implementations that send requests containing multipart message bodies MUST send a session description as a non-multipart message body if the remote implementation requests

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 669
Keysight, EX-1045
Page 669

this through an Accept header field that does not contain multipart.

SIP messages MAY contain binary bodies or body parts. When no explicit charset parameter is provided by the sender, media subtypes of the "text" type are defined to have a default charset value of "UTF-8".

### 7.4.2 Message Body Length

The body length in bytes is provided by the Content-Length header field. Section 20.14 describes the necessary contents of this header field in detail.

The "chunked" transfer encoding of HTTP/1.1 MUST NOT be used for SIP. (Note: The chunked encoding modifies the body of a message in order to transfer it as a series of chunks, each with its own size indicator.)





## 7.5 Framing SIP Messages

Unlike HTTP, SIP implementations can use UDP or other unreliable datagram protocols. Each such datagram carries one request or response. See Section 18 on constraints on usage of unreliable transports.

Implementations processing SIP messages over stream-oriented transports MUST ignore any CRLF appearing before the start-line [H4.1].

> The Content-Length header field value is used to locate the end of each SIP message in a stream. It will always be present when SIP messages are sent over stream-oriented transports.

# 8 General User Agent Behavior

A user agent represents an end system. It contains a user agent client (UAC), which generates requests, and a user agent server (UAS), which responds to them. A UAC is capable of generating a request based on some external stimulus (the user clicking a button, or a signal on a PSTN line) and processing a response. A UAS is capable of receiving a request and generating a response based on user input, external stimulus, the result of a program execution, or some other mechanism.

When a UAC sends a request, the request passes through some number of proxy servers, which forward the request towards the UAS. When the UAS generates a response, the response is forwarded towards the UAC.

UAC and UAS procedures depend strongly on two factors. First, based on whether the request or response is inside or outside of a dialog, and second, based on the method of a request. Dialogs are discussed thoroughly in Section 12; they represent a peer-to-peer relationship

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

between user agents and are established by specific SIP methods, such as INVITE.

In this section, we discuss the method-independent rules for UAC and UAS behavior when processing requests that are outside of a dialog. This includes, of course, the requests which themselves establish a dialog.

Security procedures for requests and responses outside of a dialog are described in Section 26. Specifically, mechanisms exist for the UAS and UAC to mutually authenticate. A limited set of privacy features are also supported through encryption of bodies using S/MIME.





## 8.1 UAC Behavior

This section covers UAC behavior outside of a dialog.

### 8.1.1 Generating the Request

A valid SIP request formulated by a UAC MUST, at a minimum, contain the following header fields: To, From, CSeq, Call-ID, Max-Forwards, and Via; all of these header fields are mandatory in all SIP requests. These six header fields are the fundamental building blocks of a SIP message, as they jointly provide for most of the critical message routing services including the addressing of messages, the routing of responses, limiting message propagation, ordering of messages, and the unique identification of transactions. These header fields are in addition to the mandatory request line, which contains the method, Request-URI, and SIP version.

Examples of requests sent outside of a dialog include an INVITE to establish a session (Section 13) and an OPTIONS to query for capabilities (Section 11).

#### 8.1.1.1 Request-URI

The initial Request-URI of the message SHOULD be set to the value of the URI in the To field. One notable exception is the REGISTER method; behavior for setting the Request-URI of REGISTER is given in Section 10. It may also be undesirable for privacy reasons or convenience to set these fields to the same value (especially if the originating UA expects that the Request-URI will be changed during transit).

In some special circumstances, the presence of a pre-existing route set can affect the Request-URI of the message. A pre-existing route set is an ordered set of URIs that identify a chain of servers, to which a UAC will send outgoing requests that are outside of a dialog. Commonly, they are configured on the UA by a user or service provider manually, or through some other non-SIP mechanism. When a provider wishes to configure a UA with an outbound proxy, it is RECOMMENDED that this be done by providing it with a pre-existing route set with a single URI, that of the outbound proxy.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 671
Keysight, EX-1045
Page 671

When a pre-existing route set is present, the procedures for populating the Request-URI and Route header field detailed in Section 12.2.1.1 MUST be followed (even though there is no dialog), using the desired Request-URI as the remote target URI.





8.1.1.2 To

The To header field first and foremost specifies the desired "logical" recipient of the request, or the address-of-record of the user or resource that is the target of this request. This may or may not be the ultimate recipient of the request. The To header field MAY contain a SIP or SIPS URI, but it may also make use of other URI schemes (the tel URL (RFC 2806 [9]), for example) when appropriate. All SIP implementations MUST support the SIP URI scheme. Any implementation that supports TLS MUST support the SIPS URI scheme. The To header field allows for a display name.

A UAC may learn how to populate the To header field for a particular request in a number of ways. Usually the user will suggest the To header field through a human interface, perhaps inputting the URI manually or selecting it from some sort of address book. Frequently, the user will not enter a complete URI, but rather a string of digits or letters (for example, "bob"). It is at the discretion of the UA to choose how to interpret this input. Using the string to form the user part of a SIP URI implies that the UA wishes the name to be resolved in the domain to the right-hand side (RHS) of the at-sign in the SIP URI (for instance, sip:bob@example.com). Using the string to form the user part of a SIPS URI implies that the UA wishes to communicate securely, and that the name is to be resolved in the domain to the RHS of the at-sign. The RHS will frequently be the home domain of the requestor, which allows for the home domain to process the outgoing request. This is useful for features like "speed dial" that require interpretation of the user part in the home domain. The tel URL may be used when the UA does not wish to specify the domain that should interpret a telephone number that has been input by the user. Rather, each domain through which the request passes would be given that opportunity. As an example, a user in an airport might log in and send requests through an outbound proxy in the airport. If they enter "411" (this is the phone number for local directory assistance in the United States), that needs to be interpreted and processed by the outbound proxy in the airport, not the user's home domain. In this case, tel:411 would be the right choice.

A request outside of a dialog MUST NOT contain a To tag; the tag in the To field of a request identifies the peer of the dialog. Since no dialog is established, no tag is present.

For further information on the To header field, see Section 20.39. The following is an example of a valid To header field:

```
To: Carol <sip:carol@chicago.com>
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 672
Keysight, EX-1045
Page 672

8.1.1.3 From

The From header field indicates the logical identity of the initiator of the request, possibly the user's address-of-record. Like the To header field, it contains a URI and optionally a display name. It is used by SIP elements to determine which processing rules to apply to a request (for example, automatic call rejection). As such, it is very important that the From URI not contain IP addresses or the FQDN of the host on which the UA is running, since these are not logical names.

The From header field allows for a display name. A UAC SHOULD use the display name "Anonymous", along with a syntactically correct, but otherwise meaningless URI (like sip:thisis@anonymous.invalid), if the identity of the client is to remain hidden.

Usually, the value that populates the From header field in requests generated by a particular UA is pre-provisioned by the user or by the administrators of the user's local domain. If a particular UA is used by multiple users, it might have switchable profiles that include a URI corresponding to the identity of the profiled user. Recipients of requests can authenticate the originator of a request in order to ascertain that they are who their From header field claims they are (see Section 22 for more on authentication).

The From field MUST contain a new "tag" parameter, chosen by the UAC. See Section 19.3 for details on choosing a tag.

For further information on the From header field, see Section 20.20. Examples:

```
From: "Bob" <sips:bob@biloxi.com> ;tag=a48s
From: sip:+12125551212@phone2net.com;tag=887s
From: Anonymous <sip:c8oqz84zk7z@privacy.org>;tag=hyh8
```

8.1.1.4 Call-ID

The Call-ID header field acts as a unique identifier to group together a series of messages. It MUST be the same for all requests and responses sent by either UA in a dialog. It SHOULD be the same in each registration from a UA.

In a new request created by a UAC outside of any dialog, the Call-ID header field MUST be selected by the UAC as a globally unique identifier over space and time unless overridden by method-specific behavior. All SIP UAs must have a means to guarantee that the Call-ID header fields they produce will not be inadvertently generated by any other UA. Note that when requests are retried after certain



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 673
Keysight, EX-1045
Page 673

failure responses that solicit an amendment to a request (for example, a challenge for authentication), these retried requests are not considered new requests, and therefore do not need new Call-ID header fields; see Section 8.1.3.5.

Use of cryptographically random identifiers (RFC 1750 [12]) in the generation of Call-IDs is RECOMMENDED. Implementations MAY use the form "localid@host". Call-IDs are case-sensitive and are simply compared byte-by-byte.

Using cryptographically random identifiers provides some protection against session hijacking and reduces the likelihood of unintentional Call-ID collisions.

No provisioning or human interface is required for the selection of the Call-ID header field value for a request.

For further information on the Call-ID header field, see Section 20.8.

Example:

```
Call-ID: f81d4fae-7dec-11d0-a765-00a0c91e6bf6@foo.bar.com
```

#### 8.1.1.5 CSeq

The CSeq header field serves as a way to identify and order transactions. It consists of a sequence number and a method. The method MUST match that of the request. For non-REGISTER requests outside of a dialog, the sequence number value is arbitrary. The sequence number value MUST be expressible as a 32-bit unsigned integer and MUST be less than 2**31. As long as it follows the above guidelines, a client may use any mechanism it would like to select CSeq header field values.

Section 12.2.1.1 discusses construction of the CSeq for requests within a dialog.

Example:

```
CSeq: 4711 INVITE
```



#### 8.1.1.6 Max-Forwards

The Max-Forwards header field serves to limit the number of hops a request can transit on the way to its destination. It consists of an integer that is decremented by one at each hop. If the Max-Forwards value reaches 0 before the request reaches its destination, it will

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 674
Keysight, EX-1045
Page 674

be rejected with a 483(Too Many Hops) error response.

A UAC MUST insert a Max-Forwards header field into each request it originates with a value that SHOULD be 70. This number was chosen to be sufficiently large to guarantee that a request would not be dropped in any SIP network when there were no loops, but not so large as to consume proxy resources when a loop does occur. Lower values should be used with caution and only in networks where topologies are known by the UA.

8.1.1.7 Via

The Via header field indicates the transport used for the transaction and identifies the location where the response is to be sent. A Via header field value is added only after the transport that will be used to reach the next hop has been selected (which may involve the usage of the procedures in [4]).

When the UAC creates a request, it MUST insert a Via into that request. The protocol name and protocol version in the header field MUST be SIP and 2.0, respectively. The Via header field value MUST contain a branch parameter. This parameter is used to identify the transaction created by that request. This parameter is used by both the client and the server.

The branch parameter value MUST be unique across space and time for all requests sent by the UA. The exceptions to this rule are CANCEL and ACK for non-2xx responses. As discussed below, a CANCEL request will have the same value of the branch parameter as the request it cancels. As discussed in Section 17.1.1.3, an ACK for a non-2xx response will also have the same branch ID as the INVITE whose response it acknowledges.

> The uniqueness property of the branch ID parameter, to facilitate its use as a transaction ID, was not part of RFC 2543.

The branch ID inserted by an element compliant with this specification MUST always begin with the characters "z9hG4bK". These 7 characters are used as a magic cookie (7 is deemed sufficient to ensure that an older RFC 2543 implementation would not pick such a value), so that servers receiving the request can determine that the branch ID was constructed in the fashion described by this



specification (that is, globally unique). Beyond this requirement, the precise format of the branch token is implementation-defined.

The Via header maddr, ttl, and sent-by components will be set when the request is processed by the transport layer (Section 18).

Via processing for proxies is described in Section 16.6 Item 8 and Section 16.7 Item 3.

8.1.1.8 Contact

The Contact header field provides a SIP or SIPS URI that can be used to contact that specific instance of the UA for subsequent requests.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 675
Keysight, EX-1045
Page 675

The Contact header field MUST be present and contain exactly one SIP or SIPS URI in any request that can result in the establishment of a dialog. For the methods defined in this specification, that includes only the INVITE request. For these requests, the scope of the Contact is global. That is, the Contact header field value contains the URI at which the UA would like to receive requests, and this URI MUST be valid even if used in subsequent requests outside of any dialogs.

If the Request-URI or top Route header field value contains a SIPS URI, the Contact header field MUST contain a SIPS URI as well.

For further information on the Contact header field, see Section 20.10.

8.1.1.9 Supported and Require

If the UAC supports extensions to SIP that can be applied by the server to the response, the UAC SHOULD include a Supported header field in the request listing the option tags (Section 19.2) for those extensions.

The option tags listed MUST only refer to extensions defined in standards-track RFCs. This is to prevent servers from insisting that clients implement non-standard, vendor-defined features in order to receive service. Extensions defined by experimental and informational RFCs are explicitly excluded from usage with the Supported header field in a request, since they too are often used to document vendor-defined extensions.

If the UAC wishes to insist that a UAS understand an extension that the UAC will apply to the request in order to process the request, it MUST insert a Require header field into the request listing the option tag for that extension. If the UAC wishes to apply an extension to the request and insist that any proxies that are





traversed understand that extension, it MUST insert a Proxy-Require header field into the request listing the option tag for that extension.

As with the Supported header field, the option tags in the Require and Proxy-Require header fields MUST only refer to extensions defined in standards-track RFCs.

8.1.1.10 Additional Message Components

After a new request has been created, and the header fields described above have been properly constructed, any additional optional header fields are added, as are any header fields specific to the method.

SIP requests MAY contain a MIME-encoded message-body. Regardless of the type of body that a request contains, certain header fields must be formulated to characterize the contents of the body. For further information on these header fields, see Sections 20.11 through 20.15.

8.1.2 Sending the Request

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 676
Keysight, EX-1045
Page 676

The destination for the request is then computed. Unless there is local policy specifying otherwise, the destination MUST be determined by applying the DNS procedures described in [4] as follows. If the first element in the route set indicated a strict router (resulting in forming the request as described in Section 12.2.1.1), the procedures MUST be applied to the Request-URI of the request. Otherwise, the procedures are applied to the first Route header field value in the request (if one exists), or to the request's Request-URI if there is no Route header field present. These procedures yield an ordered set of address, port, and transports to attempt. Independent of which URI is used as input to the procedures of [4], if the Request-URI specifies a SIPS resource, the UAC MUST follow the procedures of [4] as if the input URI were a SIPS URI.

Local policy MAY specify an alternate set of destinations to attempt. If the Request-URI contains a SIPS URI, any alternate destinations MUST be contacted with TLS. Beyond that, there are no restrictions on the alternate destinations if the request contains no Route header field. This provides a simple alternative to a pre-existing route set as a way to specify an outbound proxy. However, that approach for configuring an outbound proxy is NOT RECOMMENDED; a pre-existing route set with a single URI SHOULD be used instead. If the request contains a Route header field, the request SHOULD be sent to the locations derived from its topmost value, but MAY be sent to any server that the UA is certain will honor the Route and Request-URI policies specified in this document (as opposed to those in RFC 2543). In particular, a UAC configured with an outbound proxy SHOULD





attempt to send the request to the location indicated in the first Route header field value instead of adopting the policy of sending all messages to the outbound proxy.

> This ensures that outbound proxies that do not add Record-Route header field values will drop out of the path of subsequent requests. It allows endpoints that cannot resolve the first Route URI to delegate that task to an outbound proxy.

The UAC SHOULD follow the procedures defined in [4] for stateful elements, trying each address until a server is contacted. Each try constitutes a new transaction, and therefore each carries a different topmost Via header field value with a new branch parameter. Furthermore, the transport value in the Via header field is set to whatever transport was determined for the target server.

#### 8.1.3 Processing Responses

Responses are first processed by the transport layer and then passed up to the transaction layer. The transaction layer performs its processing and then passes the response up to the TU. The majority of response processing in the TU is method specific. However, there are some general behaviors independent of the method.

##### 8.1.3.1 Transaction Layer Errors

In some cases, the response returned by the transaction layer will

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 677
Keysight, EX-1045
Page 677

not be a SIP message, but rather a transaction layer error. When a timeout error is received from the transaction layer, it MUST be treated as if a 408 (Request Timeout) status code has been received. If a fatal transport error is reported by the transport layer (generally, due to fatal ICMP errors in UDP or connection failures in TCP), the condition MUST be treated as a 503 (Service Unavailable) status code.

8.1.3.2 Unrecognized Responses

A UAC MUST treat any final response it does not recognize as being equivalent to the x00 response code of that class, and MUST be able to process the x00 response code for all classes. For example, if a UAC receives an unrecognized response code of 431, it can safely assume that there was something wrong with its request and treat the response as if it had received a 400 (Bad Request) response code. A UAC MUST treat any provisional response different than 100 that it does not recognize as 183 (Session Progress). A UAC MUST be able to process 100 and 183 responses.





8.1.3.3 Vias

If more than one Via header field value is present in a response, the UAC SHOULD discard the message.

The presence of additional Via header field values that precede the originator of the request suggests that the message was misrouted or possibly corrupted.

8.1.3.4 Processing 3xx Responses

Upon receipt of a redirection response (for example, a 301 response status code), clients SHOULD use the URI(s) in the Contact header field to formulate one or more new requests based on the redirected request. This process is similar to that of a proxy recursing on a 3xx class response as detailed in Sections 16.5 and 16.6. A client starts with an initial target set containing exactly one URI, the Request-URI of the original request. If a client wishes to formulate new requests based on a 3xx class response to that request, it places the URIs to try into the target set. Subject to the restrictions in this specification, a client can choose which Contact URIs it places into the target set. As with proxy recursion, a client processing 3xx class responses MUST NOT add any given URI to the target set more than once. If the original request had a SIPS URI in the Request-URI, the client MAY choose to recurse to a non-SIPS URI, but SHOULD inform the user of the redirection to an insecure URI.

Any new request may receive 3xx responses themselves containing the original URI as a contact. Two locations can be configured to redirect to each other. Placing any given URI in the target set only once prevents infinite redirection loops.

As the target set grows, the client MAY generate new requests to the URIs in any order. A common mechanism is to order the set by the "q"

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 678
Keysight, EX-1045
Page 678

parameter value from the Contact header field value. Requests to the URIs MAY be generated serially or in parallel. One approach is to process groups of decreasing q-values serially and process the URIs in each q-value group in parallel. Another is to perform only serial processing in decreasing q-value order, arbitrarily choosing between contacts of equal q-value.

If contacting an address in the list results in a failure, as defined in the next paragraph, the element moves to the next address in the list, until the list is exhausted. If the list is exhausted, then the request has failed.





Failures SHOULD be detected through failure response codes (codes greater than 399); for network errors the client transaction will report any transport layer failures to the transaction user. Note that some response codes (detailed in 8.1.3.5) indicate that the request can be retried; requests that are reattempted should not be considered failures.

When a failure for a particular contact address is received, the client SHOULD try the next contact address. This will involve creating a new client transaction to deliver a new request.

In order to create a request based on a contact address in a 3xx response, a UAC MUST copy the entire URI from the target set into the Request-URI, except for the "method-param" and "header" URI parameters (see Section 19.1.1 for a definition of these parameters). It uses the "header" parameters to create header field values for the new request, overwriting header field values associated with the redirected request in accordance with the guidelines in Section 19.1.5.

Note that in some instances, header fields that have been communicated in the contact address may instead append to existing request header fields in the original redirected request. As a general rule, if the header field can accept a comma-separated list of values, then the new header field value MAY be appended to any existing values in the original redirected request. If the header field does not accept multiple values, the value in the original redirected request MAY be overwritten by the header field value communicated in the contact address. For example, if a contact address is returned with the following value:

```
sip:user@host?Subject=foo&Call-Info=<http://www.foo.com>
```

Then any Subject header field in the original redirected request is overwritten, but the HTTP URL is merely appended to any existing Call-Info header field values.

It is RECOMMENDED that the UAC reuse the same To, From, and Call-ID used in the original redirected request, but the UAC MAY also choose to update the Call-ID header field value for new requests, for example.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 679
Keysight, EX-1045
Page 679

Finally, once the new request has been constructed, it is sent using a new client transaction, and therefore MUST have a new branch ID in the top Via field as discussed in Section 8.1.1.7.



In all other respects, requests sent upon receipt of a redirect response SHOULD re-use the header fields and bodies of the original request.

In some instances, Contact header field values may be cached at UAC temporarily or permanently depending on the status code received and the presence of an expiration interval; see Sections 21.3.2 and 21.3.3.

#### 8.1.3.5 Processing 4xx Responses

Certain 4xx response codes require specific UA processing, independent of the method.

If a 401 (Unauthorized) or 407 (Proxy Authentication Required) response is received, the UAC SHOULD follow the authorization procedures of Section 22.2 and Section 22.3 to retry the request with credentials.

If a 413 (Request Entity Too Large) response is received (Section 21.4.11), the request contained a body that was longer than the UAS was willing to accept. If possible, the UAC SHOULD retry the request, either omitting the body or using one of a smaller length.

If a 415 (Unsupported Media Type) response is received (Section 21.4.13), the request contained media types not supported by the UAS. The UAC SHOULD retry sending the request, this time only using content with types listed in the Accept header field in the response, with encodings listed in the Accept-Encoding header field in the response, and with languages listed in the Accept-Language in the response.

If a 416 (Unsupported URI Scheme) response is received (Section 21.4.14), the Request-URI used a URI scheme not supported by the server. The client SHOULD retry the request, this time, using a SIP URI.

If a 420 (Bad Extension) response is received (Section 21.4.15), the request contained a Require or Proxy-Require header field listing an option-tag for a feature not supported by a proxy or UAS. The UAC SHOULD retry the request, this time omitting any extensions listed in the Unsupported header field in the response.

In all of the above cases, the request is retried by creating a new request with the appropriate modifications. This new request constitutes a new transaction and SHOULD have the same value of the Call-ID, To, and From of the previous request, but the CSeq should contain a new sequence number that is one higher than the previous.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

With other 4xx responses, including those yet to be defined, a retry may or may not be possible depending on the method and the use case.

### 8.2 UAS Behavior

When a request outside of a dialog is processed by a UAS, there is a set of processing rules that are followed, independent of the method. Section 12 gives guidance on how a UAS can tell whether a request is inside or outside of a dialog.

Note that request processing is atomic. If a request is accepted, all state changes associated with it MUST be performed. If it is rejected, all state changes MUST NOT be performed.

UASs SHOULD process the requests in the order of the steps that follow in this section (that is, starting with authentication, then inspecting the method, the header fields, and so on throughout the remainder of this section).

#### 8.2.1 Method Inspection

Once a request is authenticated (or authentication is skipped), the UAS MUST inspect the method of the request. If the UAS recognizes but does not support the method of a request, it MUST generate a 405 (Method Not Allowed) response. Procedures for generating responses are described in Section 8.2.6. The UAS MUST also add an Allow header field to the 405 (Method Not Allowed) response. The Allow header field MUST list the set of methods supported by the UAS generating the message. The Allow header field is presented in Section 20.5.

If the method is one supported by the server, processing continues.

#### 8.2.2 Header Inspection

If a UAS does not understand a header field in a request (that is, the header field is not defined in this specification or in any supported extension), the server MUST ignore that header field and continue processing the message. A UAS SHOULD ignore any malformed header fields that are not necessary for processing requests.

##### 8.2.2.1 To and Request-URI

The To header field identifies the original recipient of the request designated by the user identified in the From field. The original recipient may or may not be the UAS processing the request, due to call forwarding or other proxy operations. A UAS MAY apply any policy it wishes to determine whether to accept requests when the To



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 681
Keysight, EX-1045
Page 681

header field is not the identity of the UAS. However, it is RECOMMENDED that a UAS accept requests even if they do not recognize the URI scheme (for example, a tel: URI) in the To header field, or if the To header field does not address a known or current user of this UAS. If, on the other hand, the UAS decides to reject the request, it SHOULD generate a response with a 403 (Forbidden) status code and pass it to the server transaction for transmission.

However, the Request-URI identifies the UAS that is to process the request. If the Request-URI uses a scheme not supported by the UAS, it SHOULD reject the request with a 416 (Unsupported URI Scheme) response. If the Request-URI does not identify an address that the UAS is willing to accept requests for, it SHOULD reject the request with a 404 (Not Found) response. Typically, a UA that uses the REGISTER method to bind its address-of-record to a specific contact address will see requests whose Request-URI equals that contact address. Other potential sources of received Request-URIs include the Contact header fields of requests and responses sent by the UA that establish or refresh dialogs.

8.2.2.2 Merged Requests

If the request has no tag in the To header field, the UAS core MUST check the request against ongoing transactions. If the From tag, Call-ID, and CSeq exactly match those associated with an ongoing transaction, but the request does not match that transaction (based on the matching rules in Section 17.2.3), the UAS core SHOULD generate a 482 (Loop Detected) response and pass it to the server transaction.

> The same request has arrived at the UAS more than once, following different paths, most likely due to forking. The UAS processes the first such request received and responds with a 482 (Loop Detected) to the rest of them.

8.2.2.3 Require

Assuming the UAS decides that it is the proper element to process the request, it examines the Require header field, if present.

The Require header field is used by a UAC to tell a UAS about SIP extensions that the UAC expects the UAS to support in order to process the request properly. Its format is described in Section 20.32. If a UAS does not understand an option-tag listed in a Require header field, it MUST respond by generating a response with status code 420 (Bad Extension). The UAS MUST add an Unsupported header field, and list in it those options it does not understand amongst those in the Require header field of the request.



Note that Require and Proxy-Require MUST NOT be used in a SIP CANCEL request, or in an ACK request sent for a non-2xx response. These header fields MUST be ignored if they are present in these requests.

An ACK request for a 2xx response MUST contain only those Require and Proxy-Require values that were present in the initial request.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 682
Keysight, EX-1045
Page 682

Example:

```
UAC->UAS:   INVITE sip:watson@bell-telephone.com SIP/2.0
            Require: 100rel

UAS->UAC:   SIP/2.0 420 Bad Extension
            Unsupported: 100rel
```

This behavior ensures that the client-server interaction will proceed without delay when all options are understood by both sides, and only slow down if options are not understood (as in the example above). For a well-matched client-server pair, the interaction proceeds quickly, saving a round-trip often required by negotiation mechanisms. In addition, it also removes ambiguity when the client requires features that the server does not understand. Some features, such as call handling fields, are only of interest to end systems.

#### 8.2.3 Content Processing

Assuming the UAS understands any extensions required by the client, the UAS examines the body of the message, and the header fields that describe it. If there are any bodies whose type (indicated by the Content-Type), language (indicated by the Content-Language) or encoding (indicated by the Content-Encoding) are not understood, and that body part is not optional (as indicated by the Content-Disposition header field), the UAS MUST reject the request with a 415 (Unsupported Media Type) response. The response MUST contain an Accept header field listing the types of all bodies it understands, in the event the request contained bodies of types not supported by the UAS. If the request contained content encodings not understood by the UAS, the response MUST contain an Accept-Encoding header field listing the encodings understood by the UAS. If the request contained content with languages not understood by the UAS, the response MUST contain an Accept-Language header field indicating the languages understood by the UAS. Beyond these checks, body handling depends on the method and type. For further information on the processing of content-specific header fields, see Section 7.4 as well as Section 20.11 through 20.15.



#### 8.2.4 Applying Extensions

A UAS that wishes to apply some extension when generating the response MUST NOT do so unless support for that extension is indicated in the Supported header field in the request. If the desired extension is not supported, the server SHOULD rely only on baseline SIP and any other extensions supported by the client. In rare circumstances, where the server cannot process the request without the extension, the server MAY send a 421 (Extension Required) response. This response indicates that the proper response cannot be generated without support of a specific extension. The needed extension(s) MUST be included in a Require header field in the response. This behavior is NOT RECOMMENDED, as it will generally

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 683
Keysight, EX-1045
Page 683

break interoperability.

Any extensions applied to a non-421 response MUST be listed in a Require header field included in the response. Of course, the server MUST NOT apply extensions not listed in the Supported header field in the request. As a result of this, the Require header field in a response will only ever contain option tags defined in standards-track RFCs.

8.2.5 Processing the Request

Assuming all of the checks in the previous subsections are passed, the UAS processing becomes method-specific. Section 10 covers the REGISTER request, Section 11 covers the OPTIONS request, Section 13 covers the INVITE request, and Section 15 covers the BYE request.

8.2.6 Generating the Response

When a UAS wishes to construct a response to a request, it follows the general procedures detailed in the following subsections. Additional behaviors specific to the response code in question, which are not detailed in this section, may also be required.

Once all procedures associated with the creation of a response have been completed, the UAS hands the response back to the server transaction from which it received the request.

8.2.6.1 Sending a Provisional Response

One largely non-method-specific guideline for the generation of responses is that UASs SHOULD NOT issue a provisional response for a non-INVITE request. Rather, UASs SHOULD generate a final response to a non-INVITE request as soon as possible.





When a 100 (Trying) response is generated, any Timestamp header field present in the request MUST be copied into this 100 (Trying) response. If there is a delay in generating the response, the UAS SHOULD add a delay value into the Timestamp value in the response. This value MUST contain the difference between the time of sending of the response and receipt of the request, measured in seconds.

8.2.6.2 Headers and Tags

The From field of the response MUST equal the From header field of the request. The Call-ID header field of the response MUST equal the Call-ID header field of the request. The CSeq header field of the response MUST equal the CSeq field of the request. The Via header field values in the response MUST equal the Via header field values in the request and MUST maintain the same ordering.

If a request contained a To tag in the request, the To header field in the response MUST equal that of the request. However, if the To header field in the request did not contain a tag, the URI in the To header field in the response MUST equal the URI in the To header

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

field; additionally, the UAS MUST add a tag to the To header field in the response (with the exception of the 100 (Trying) response, in which a tag MAY be present). This serves to identify the UAS that is responding, possibly resulting in a component of a dialog ID. The same tag MUST be used for all responses to that request, both final and provisional (again excepting the 100 (Trying)). Procedures for the generation of tags are defined in Section 19.3.

### 8.2.7 Stateless UAS Behavior

A stateless UAS is a UAS that does not maintain transaction state. It replies to requests normally, but discards any state that would ordinarily be retained by a UAS after a response has been sent. If a stateless UAS receives a retransmission of a request, it regenerates the response and resends it, just as if it were replying to the first instance of the request. A UAS cannot be stateless unless the request processing for that method would always result in the same response if the requests are identical. This rules out stateless registrars, for example. Stateless UASs do not use a transaction layer; they receive requests directly from the transport layer and send responses directly to the transport layer.

The stateless UAS role is needed primarily to handle unauthenticated requests for which a challenge response is issued. If unauthenticated requests were handled statefully, then malicious floods of unauthenticated requests could create massive amounts of





transaction state that might slow or completely halt call processing in a UAS, effectively creating a denial of service condition; for more information see Section 26.1.5.

The most important behaviors of a stateless UAS are the following:

- A stateless UAS MUST NOT send provisional (1xx) responses.
- A stateless UAS MUST NOT retransmit responses.
- A stateless UAS MUST ignore ACK requests.
- A stateless UAS MUST ignore CANCEL requests.
- To header tags MUST be generated for responses in a stateless manner - in a manner that will generate the same tag for the same request consistently. For information on tag construction see Section 19.3.

In all other respects, a stateless UAS behaves in the same manner as a stateful UAS. A UAS can operate in either a stateful or stateless mode for each new request.

## 8.3 Redirect Servers

In some architectures it may be desirable to reduce the processing load on proxy servers that are responsible for routing requests, and

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 685
Keysight, EX-1045
Page 685

improve signaling path robustness, by relying on redirection.

Redirection allows servers to push routing information for a request back in a response to the client, thereby taking themselves out of the loop of further messaging for this transaction while still aiding in locating the target of the request. When the originator of the request receives the redirection, it will send a new request based on the URI(s) it has received. By propagating URIs from the core of the network to its edges, redirection allows for considerable network scalability.

A redirect server is logically constituted of a server transaction layer and a transaction user that has access to a location service of some kind (see Section 10 for more on registrars and location services). This location service is effectively a database containing mappings between a single URI and a set of one or more alternative locations at which the target of that URI can be found.

A redirect server does not issue any SIP requests of its own. After receiving a request other than CANCEL, the server either refuses the request or gathers the list of alternative locations from the





location service and returns a final response of class 3xx. For well-formed CANCEL requests, it SHOULD return a 2xx response. This response ends the SIP transaction. The redirect server maintains transaction state for an entire SIP transaction. It is the responsibility of clients to detect forwarding loops between redirect servers.

When a redirect server returns a 3xx response to a request, it populates the list of (one or more) alternative locations into the Contact header field. An "expires" parameter to the Contact header field values may also be supplied to indicate the lifetime of the Contact data.

The Contact header field contains URIs giving the new locations or user names to try, or may simply specify additional transport parameters. A 301 (Moved Permanently) or 302 (Moved Temporarily) response may also give the same location and username that was targeted by the initial request but specify additional transport parameters such as a different server or multicast address to try, or a change of SIP transport from UDP to TCP or vice versa.

However, redirect servers MUST NOT redirect a request to a URI equal to the one in the Request-URI; instead, provided that the URI does not point to itself, the server MAY proxy the request to the destination URI, or MAY reject it with a 404.

> If a client is using an outbound proxy, and that proxy actually redirects requests, a potential arises for infinite redirection loops.

Note that a Contact header field value MAY also refer to a different resource than the one originally called. For example, a SIP call connected to PSTN gateway may need to deliver a special informational announcement such as "The number you have dialed has been changed."

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 686
Keysight, EX-1045
Page 686

A Contact response header field can contain any suitable URI indicating where the called party can be reached, not limited to SIP URIs. For example, it could contain URIs for phones, fax, or irc (if they were defined) or a mailto: (RFC 2368 [32]) URL. Section 26.4.4 discusses implications and limitations of redirecting a SIPS URI to a non-SIPS URI.

The "expires" parameter of a Contact header field value indicates how long the URI is valid. The value of the parameter is a number indicating seconds. If this parameter is not provided, the value of the Expires header field determines how long the URI is valid. Malformed values SHOULD be treated as equivalent to 3600.





This provides a modest level of backwards compatibility with RFC 2543, which allowed absolute times in this header field. If an absolute time is received, it will be treated as malformed, and then default to 3600.

Redirect servers MUST ignore features that are not understood (including unrecognized header fields, any unknown option tags in Require, or even method names) and proceed with the redirection of the request in question.

## 9 Canceling a Request

The previous section has discussed general UA behavior for generating requests and processing responses for requests of all methods. In this section, we discuss a general purpose method, called CANCEL.

The CANCEL request, as the name implies, is used to cancel a previous request sent by a client. Specifically, it asks the UAS to cease processing the request and to generate an error response to that request. CANCEL has no effect on a request to which a UAS has already given a final response. Because of this, it is most useful to CANCEL requests to which it can take a server long time to respond. For this reason, CANCEL is best for INVITE requests, which can take a long time to generate a response. In that usage, a UAS that receives a CANCEL request for an INVITE, but has not yet sent a final response, would "stop ringing", and then respond to the INVITE with a specific error response (a 487).

CANCEL requests can be constructed and sent by both proxies and user agent clients. Section 15 discusses under what conditions a UAC would CANCEL an INVITE request, and Section 16.10 discusses proxy usage of CANCEL.

A stateful proxy responds to a CANCEL, rather than simply forwarding a response it would receive from a downstream element. For that reason, CANCEL is referred to as a "hop-by-hop" request, since it is responded to at each stateful proxy hop.

### 9.1 Client Behavior

A CANCEL request SHOULD NOT be sent to cancel a request other than

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 687
Keysight, EX-1045
Page 687

INVITE.

Since requests other than INVITE are responded to immediately, sending a CANCEL for a non-INVITE request would always create a race condition.





The following procedures are used to construct a CANCEL request. The Request-URI, Call-ID, To, the numeric part of CSeq, and From header fields in the CANCEL request MUST be identical to those in the request being cancelled, including tags. A CANCEL constructed by a client MUST have only a single Via header field value matching the top Via value in the request being cancelled. Using the same values for these header fields allows the CANCEL to be matched with the request it cancels (Section 9.2 indicates how such matching occurs). However, the method part of the CSeq header field MUST have a value of CANCEL. This allows it to be identified and processed as a transaction in its own right (See Section 17).

If the request being cancelled contains a Route header field, the CANCEL request MUST include that Route header field's values.

This is needed so that stateless proxies are able to route CANCEL requests properly.

The CANCEL request MUST NOT contain any Require or Proxy-Require header fields.

Once the CANCEL is constructed, the client SHOULD check whether it has received any response (provisional or final) for the request being cancelled (herein referred to as the "original request").

If no provisional response has been received, the CANCEL request MUST NOT be sent; rather, the client MUST wait for the arrival of a provisional response before sending the request. If the original request has generated a final response, the CANCEL SHOULD NOT be sent, as it is an effective no-op, since CANCEL has no effect on requests that have already generated a final response. When the client decides to send the CANCEL, it creates a client transaction for the CANCEL and passes it the CANCEL request along with the destination address, port, and transport. The destination address, port, and transport for the CANCEL MUST be identical to those used to send the original request.

If it was allowed to send the CANCEL before receiving a response for the previous request, the server could receive the CANCEL before the original request.

Note that both the transaction corresponding to the original request and the CANCEL transaction will complete independently. However, a UAC canceling a request cannot rely on receiving a 487 (Request Terminated) response for the original request, as an RFC 2543-compliant UAS will not generate such a response. If there is no final response for the original request in 64*T1 seconds (T1 is

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 688
Keysight, EX-1045
Page 688

defined in Section 17.1.1.1), the client SHOULD then consider the original transaction cancelled and SHOULD destroy the client transaction handling the original request.

## 9.2 Server Behavior

The CANCEL method requests that the TU at the server side cancel a pending transaction. The TU determines the transaction to be cancelled by taking the CANCEL request, and then assuming that the request method is anything but CANCEL or ACK and applying the transaction matching procedures of Section 17.2.3. The matching transaction is the one to be cancelled.

The processing of a CANCEL request at a server depends on the type of server. A stateless proxy will forward it, a stateful proxy might respond to it and generate some CANCEL requests of its own, and a UAS will respond to it. See Section 16.10 for proxy treatment of CANCEL.

A UAS first processes the CANCEL request according to the general UAS processing described in Section 8.2. However, since CANCEL requests are hop-by-hop and cannot be resubmitted, they cannot be challenged by the server in order to get proper credentials in an Authorization header field. Note also that CANCEL requests do not contain a Require header field.

If the UAS did not find a matching transaction for the CANCEL according to the procedure above, it SHOULD respond to the CANCEL with a 481 (Call Leg/Transaction Does Not Exist). If the transaction for the original request still exists, the behavior of the UAS on receiving a CANCEL request depends on whether it has already sent a final response for the original request. If it has, the CANCEL request has no effect on the processing of the original request, no effect on any session state, and no effect on the responses generated for the original request. If the UAS has not issued a final response for the original request, its behavior depends on the method of the original request. If the original request was an INVITE, the UAS SHOULD immediately respond to the INVITE with a 487 (Request Terminated). A CANCEL request has no impact on the processing of transactions with any other method defined in this specification.

Regardless of the method of the original request, as long as the CANCEL matched an existing transaction, the UAS answers the CANCEL request itself with a 200 (OK) response. This response is constructed following the procedures described in Section 8.2.6 noting that the To tag of the response to the CANCEL and the To tag in the response to the original request SHOULD be the same. The response to CANCEL is passed to the server transaction for transmission.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 689
Keysight, EX-1045
Page 689

# 10 Registrations

## 10.1 Overview

SIP offers a discovery capability. If a user wants to initiate a session with another user, SIP must discover the current host(s) at which the destination user is reachable. This discovery process is frequently accomplished by SIP network elements such as proxy servers and redirect servers which are responsible for receiving a request, determining where to send it based on knowledge of the location of the user, and then sending it there. To do this, SIP network elements consult an abstract service known as a location service, which provides address bindings for a particular domain. These address bindings map an incoming SIP or SIPS URI, sip:bob@biloxi.com, for example, to one or more URIs that are somehow "closer" to the desired user, sip:bob@engineering.biloxi.com, for example. Ultimately, a proxy will consult a location service that maps a received URI to the user agent(s) at which the desired recipient is currently residing.

Registration creates bindings in a location service for a particular domain that associates an address-of-record URI with one or more contact addresses. Thus, when a proxy for that domain receives a request whose Request-URI matches the address-of-record, the proxy will forward the request to the contact addresses registered to that address-of-record. Generally, it only makes sense to register an address-of-record at a domain's location service when requests for that address-of-record would be routed to that domain. In most cases, this means that the domain of the registration will need to match the domain in the URI of the address-of-record.

There are many ways by which the contents of the location service can be established. One way is administratively. In the above example, Bob is known to be a member of the engineering department through access to a corporate database. However, SIP provides a mechanism for a UA to create a binding explicitly. This mechanism is known as registration.

Registration entails sending a REGISTER request to a special type of UAS known as a registrar. A registrar acts as the front end to the location service for a domain, reading and writing mappings based on the contents of REGISTER requests. This location service is then typically consulted by a proxy server that is responsible for routing requests for that domain.

An illustration of the overall registration process is given in Figure 2. Note that the registrar and proxy server are logical roles that can be played by a single device in a network; for purposes of



clarity the two are separated in this illustration. Also note that UAs may send requests through a proxy server in order to reach a registrar if the two are separate elements.

SIP does not mandate a particular mechanism for implementing the location service. The only requirement is that a registrar for some

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 690
Keysight, EX-1045
Page 690

domain MUST be able to read and write data to the location service, and a proxy or a redirect server for that domain MUST be capable of reading that same data. A registrar MAY be co-located with a particular SIP proxy server for the same domain.

### 10.2 Constructing the REGISTER Request

REGISTER requests add, remove, and query bindings. A REGISTER request can add a new binding between an address-of-record and one or more contact addresses. Registration on behalf of a particular address-of-record can be performed by a suitably authorized third party. A client can also remove previous bindings or query to determine which bindings are currently in place for an address-of-record.

Except as noted, the construction of the REGISTER request and the behavior of clients sending a REGISTER request is identical to the general UAC behavior described in Section 8.1 and Section 17.1.

A REGISTER request does not establish a dialog. A UAC MAY include a Route header field in a REGISTER request based on a pre-existing route set as described in Section 8.1. The Record-Route header field has no meaning in REGISTER requests or responses, and MUST be ignored if present. In particular, the UAC MUST NOT create a new route set based on the presence or absence of a Record-Route header field in any response to a REGISTER request.

The following header fields, except Contact, MUST be included in a REGISTER request. A Contact header field MAY be included:

Request-URI: The Request-URI names the domain of the location service for which the registration is meant (for example, "sip:chicago.com"). The "userinfo" and "@" components of the SIP URI MUST NOT be present.

To: The To header field contains the address of record whose registration is to be created, queried, or modified. The To header field and the Request-URI field typically differ, as the former contains a user name. This address-of-record MUST be a SIP URI or SIPS URI.





From: The From header field contains the address-of-record of the person responsible for the registration. The value is the same as the To header field unless the request is a third-party registration.

Call-ID: All registrations from a UAC SHOULD use the same Call-ID header field value for registrations sent to a particular registrar.

If the same client were to use different Call-ID values, a registrar could not detect whether a delayed REGISTER request might have arrived out of order.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 691
Keysight, EX-1045
Page 691

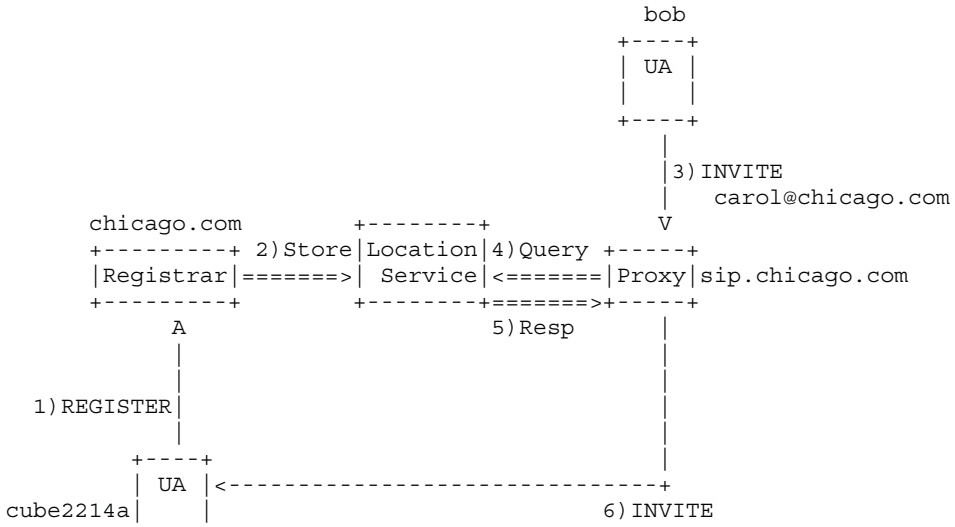
CSeq: The CSeq value guarantees proper ordering of REGISTER requests. A UA MUST increment the CSeq value by one for each REGISTER request with the same Call-ID.

Contact: REGISTER requests MAY contain a Contact header field with zero or more values containing address bindings.

UAs MUST NOT send a new registration (that is, containing new Contact header field values, as opposed to a retransmission) until they have received a final response from the registrar for the previous one or the previous REGISTER request has timed out.








https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 692
Keysight, EX-1045
Page 692

```
          +-----+                    carol@cube2214a.chicago.com
           carol
```

Figure 2: REGISTER example

The following Contact header parameters have a special meaning in REGISTER requests:

action: The "action" parameter from RFC 2543 has been deprecated. UACs SHOULD NOT use the "action" parameter.

expires: The "expires" parameter indicates how long the UA would like the binding to be valid. The value is a number indicating seconds. If this parameter is not provided, the value of the Expires header field is used instead. Implementations MAY treat values larger than 2**32-1 (4294967295 seconds or 136 years) as equivalent to 2**32-1. Malformed values SHOULD be treated as equivalent to 3600.

### 10.2.1 Adding Bindings

The REGISTER request sent to a registrar includes the contact address(es) to which SIP requests for the address-of-record should be forwarded. The address-of-record is included in the To header field of the REGISTER request.





The Contact header field values of the request typically consist of SIP or SIPS URIs that identify particular SIP endpoints (for example, "sip:carol@cube2214a.chicago.com"), but they MAY use any URI scheme. A SIP UA can choose to register telephone numbers (with the tel URL, RFC 2806 [9]) or email addresses (with a mailto URL, RFC 2368 [32]) as Contacts for an address-of-record, for example.

For example, Carol, with address-of-record "sip:carol@chicago.com", would register with the SIP registrar of the domain chicago.com. Her registrations would then be used by a proxy server in the chicago.com domain to route requests for Carol's address-of-record to her SIP endpoint.

Once a client has established bindings at a registrar, it MAY send subsequent registrations containing new bindings or modifications to existing bindings as necessary. The 2xx response to the REGISTER request will contain, in a Contact header field, a complete list of bindings that have been registered for this address-of-record at this registrar.

If the address-of-record in the To header field of a REGISTER request is a SIPS URI, then any Contact header field values in the request SHOULD also be SIPS URIs. Clients should only register non-SIPS URIs under a SIPS address-of-record when the security of the resource represented by the contact address is guaranteed by other means. This may be applicable to URIs that invoke protocols other than SIP, or SIP devices secured by protocols other than TLS.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

Registrations do not need to update all bindings. Typically, a UA only updates its own contact addresses.

#### 10.2.1.1 Setting the Expiration Interval of Contact Addresses

When a client sends a REGISTER request, it MAY suggest an expiration interval that indicates how long the client would like the registration to be valid. (As described in Section 10.3, the registrar selects the actual time interval based on its local policy.)

There are two ways in which a client can suggest an expiration interval for a binding: through an Expires header field or an "expires" Contact header parameter. The latter allows expiration intervals to be suggested on a per-binding basis when more than one binding is given in a single REGISTER request, whereas the former suggests an expiration interval for all Contact header field values that do not contain the "expires" parameter.





If neither mechanism for expressing a suggested expiration time is present in a REGISTER, the client is indicating its desire for the server to choose.

#### 10.2.1.2 Preferences among Contact Addresses

If more than one Contact is sent in a REGISTER request, the registering UA intends to associate all of the URIs in these Contact header field values with the address-of-record present in the To field. This list can be prioritized with the "q" parameter in the Contact header field. The "q" parameter indicates a relative preference for the particular Contact header field value compared to other bindings for this address-of-record. Section 16.6 describes how a proxy server uses this preference indication.

### 10.2.2 Removing Bindings

Registrations are soft state and expire unless refreshed, but can also be explicitly removed. A client can attempt to influence the expiration interval selected by the registrar as described in Section 10.2.1. A UA requests the immediate removal of a binding by specifying an expiration interval of "0" for that contact address in a REGISTER request. UAs SHOULD support this mechanism so that bindings can be removed before their expiration interval has passed.

The REGISTER-specific Contact header field value of "*" applies to all registrations, but it MUST NOT be used unless the Expires header field is present with a value of "0".

> Use of the "*" Contact header field value allows a registering UA to remove all bindings associated with an address-of-record without knowing their precise values.

### 10.2.3 Fetching Bindings

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 694
Keysight, EX-1045
Page 694

A success response to any REGISTER request contains the complete list of existing bindings, regardless of whether the request contained a Contact header field. If no Contact header field is present in a REGISTER request, the list of bindings is left unchanged.

### 10.2.4 Refreshing Bindings

Each UA is responsible for refreshing the bindings that it has previously established. A UA SHOULD NOT refresh bindings set up by other UAs.





The 200 (OK) response from the registrar contains a list of Contact fields enumerating all current bindings. The UA compares each contact address to see if it created the contact address, using comparison rules in Section 19.1.4. If so, it updates the expiration time interval according to the expires parameter or, if absent, the Expires field value. The UA then issues a REGISTER request for each of its bindings before the expiration interval has elapsed. It MAY combine several updates into one REGISTER request.

A UA SHOULD use the same Call-ID for all registrations during a single boot cycle. Registration refreshes SHOULD be sent to the same network address as the original registration, unless redirected.

### 10.2.5 Setting the Internal Clock

If the response for a REGISTER request contains a Date header field, the client MAY use this header field to learn the current time in order to set any internal clocks.

### 10.2.6 Discovering a Registrar

UAs can use three ways to determine the address to which to send registrations: by configuration, using the address-of-record, and multicast. A UA can be configured, in ways beyond the scope of this specification, with a registrar address. If there is no configured registrar address, the UA SHOULD use the host part of the address-of-record as the Request-URI and address the request there, using the normal SIP server location mechanisms [4]. For example, the UA for the user "sip:carol@chicago.com" addresses the REGISTER request to "sip:chicago.com".

Finally, a UA can be configured to use multicast. Multicast registrations are addressed to the well-known "all SIP servers" multicast address "sip.mcast.net" (224.0.1.75 for IPv4). No well-known IPv6 multicast address has been allocated; such an allocation will be documented separately when needed. SIP UAs MAY listen to that address and use it to become aware of the location of other local users (see [33]); however, they do not respond to the request.

> Multicast registration may be inappropriate in some environments, for example, if multiple businesses share the same local area

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 695
Keysight, EX-1045
Page 695

network.

10.2.7 Transmitting a Request

Once the REGISTER method has been constructed, and the destination of the message identified, UACs follow the procedures described in Section 8.1.2 to hand off the REGISTER to the transaction layer.





If the transaction layer returns a timeout error because the REGISTER yielded no response, the UAC SHOULD NOT immediately re-attempt a registration to the same registrar.

An immediate re-attempt is likely to also timeout. Waiting some reasonable time interval for the conditions causing the timeout to be corrected reduces unnecessary load on the network. No specific interval is mandated.

10.2.8 Error Responses

If a UA receives a 423 (Interval Too Brief) response, it MAY retry the registration after making the expiration interval of all contact addresses in the REGISTER request equal to or greater than the expiration interval within the Min-Expires header field of the 423 (Interval Too Brief) response.

10.3 Processing REGISTER Requests

A registrar is a UAS that responds to REGISTER requests and maintains a list of bindings that are accessible to proxy servers and redirect servers within its administrative domain. A registrar handles requests according to Section 8.2 and Section 17.2, but it accepts only REGISTER requests. A registrar MUST not generate 6xx responses.

A registrar MAY redirect REGISTER requests as appropriate. One common usage would be for a registrar listening on a multicast interface to redirect multicast REGISTER requests to its own unicast interface with a 302 (Moved Temporarily) response.

Registrars MUST ignore the Record-Route header field if it is included in a REGISTER request. Registrars MUST NOT include a Record-Route header field in any response to a REGISTER request.

A registrar might receive a request that traversed a proxy which treats REGISTER as an unknown request and which added a Record-Route header field value.

A registrar has to know (for example, through configuration) the set of domain(s) for which it maintains bindings. REGISTER requests MUST be processed by a registrar in the order that they are received. REGISTER requests MUST also be processed atomically, meaning that a particular REGISTER request is either processed completely or not at all. Each REGISTER message MUST be processed independently of any other registration or binding changes.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

When receiving a REGISTER request, a registrar follows these steps:

1. The registrar inspects the Request-URI to determine whether it has access to bindings for the domain identified in the Request-URI. If not, and if the server also acts as a proxy server, the server SHOULD forward the request to the addressed domain, following the general behavior for proxying messages described in Section 16.

2. To guarantee that the registrar supports any necessary extensions, the registrar MUST process the Require header field values as described for UASs in Section 8.2.2.

3. A registrar SHOULD authenticate the UAC. Mechanisms for the authentication of SIP user agents are described in Section 22. Registration behavior in no way overrides the generic authentication framework for SIP. If no authentication mechanism is available, the registrar MAY take the From address as the asserted identity of the originator of the request.

4. The registrar SHOULD determine if the authenticated user is authorized to modify registrations for this address-of-record. For example, a registrar might consult an authorization database that maps user names to a list of addresses-of-record for which that user has authorization to modify bindings. If the authenticated user is not authorized to modify bindings, the registrar MUST return a 403 (Forbidden) and skip the remaining steps.

   In architectures that support third-party registration, one entity may be responsible for updating the registrations associated with multiple addresses-of-record.

5. The registrar extracts the address-of-record from the To header field of the request. If the address-of-record is not valid for the domain in the Request-URI, the registrar MUST send a 404 (Not Found) response and skip the remaining steps. The URI MUST then be converted to a canonical form. To do that, all URI parameters MUST be removed (including the user-param), and any escaped characters MUST be converted to their unescaped form. The result serves as an index into the list of bindings.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 697
Keysight, EX-1045
Page 697

6. The registrar checks whether the request contains the Contact header field. If not, it skips to the last step. If the Contact header field is present, the registrar checks if there is one Contact field value that contains the special value "*" and an Expires field. If the request has additional Contact fields or an expiration time other than zero, the request is invalid, and the server MUST return a 400 (Invalid Request) and skip the remaining steps. If not, the registrar checks whether the Call-ID agrees with the value stored for each binding. If not, it MUST remove the binding. If it does agree, it MUST remove the binding only if the CSeq in the request is higher than the value stored for that binding. Otherwise, the update MUST be aborted and the request fails.

7. The registrar now processes each contact address in the Contact header field in turn. For each address, it determines the expiration interval as follows:

   - If the field value has an "expires" parameter, that value MUST be taken as the requested expiration.

   - If there is no such parameter, but the request has an Expires header field, that value MUST be taken as the requested expiration.

   - If there is neither, a locally-configured default value MUST be taken as the requested expiration.

   The registrar MAY choose an expiration less than the requested expiration interval. If and only if the requested expiration interval is greater than zero AND smaller than one hour AND less than a registrar-configured minimum, the registrar MAY reject the registration with a response of 423 (Interval Too Brief). This response MUST contain a Min-Expires header field that states the minimum expiration interval the registrar is willing to honor. It then skips the remaining steps.

   Allowing the registrar to set the registration interval protects it against excessively frequent registration refreshes while limiting the state that it needs to maintain and decreasing the likelihood of registrations going stale. The expiration interval of a registration is frequently used in the creation of services. An example is a follow-me service, where the user may only be available at a terminal for a brief period. Therefore, registrars should accept brief registrations; a request should only be rejected if the interval is so short that the refreshes would degrade registrar performance.



   For each address, the registrar then searches the list of current bindings using the URI comparison rules. If the binding does not exist, it is tentatively added. If the binding does exist, the registrar checks the Call-ID value. If the Call-ID value in the existing binding differs from the Call-ID value in the request, the binding MUST be removed if

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 698
Keysight, EX-1045
Page 698

the expiration time is zero and updated otherwise. If they are the same, the registrar compares the CSeq value. If the value is higher than that of the existing binding, it MUST update or remove the binding as above. If not, the update MUST be aborted and the request fails.

This algorithm ensures that out-of-order requests from the same UA are ignored.

Each binding record records the Call-ID and CSeq values from the request.

The binding updates MUST be committed (that is, made visible to the proxy or redirect server) if and only if all binding updates and additions succeed. If any one of them fails (for example, because the back-end database commit failed), the request MUST fail with a 500 (Server Error) response and all tentative binding updates MUST be removed.

8. The registrar returns a 200 (OK) response. The response MUST contain Contact header field values enumerating all current bindings. Each Contact value MUST feature an "expires" parameter indicating its expiration interval chosen by the registrar. The response SHOULD include a Date header field.

## 11 Querying for Capabilities

The SIP method OPTIONS allows a UA to query another UA or a proxy server as to its capabilities. This allows a client to discover information about the supported methods, content types, extensions, codecs, etc. without "ringing" the other party. For example, before a client inserts a Require header field into an INVITE listing an option that it is not certain the destination UAS supports, the client can query the destination UAS with an OPTIONS to see if this option is returned in a Supported header field. All UAs MUST support the OPTIONS method.

The target of the OPTIONS request is identified by the Request-URI, which could identify another UA or a SIP server. If the OPTIONS is addressed to a proxy server, the Request-URI is set without a user part, similar to the way a Request-URI is set for a REGISTER request.





Alternatively, a server receiving an OPTIONS request with a Max-Forwards header field value of 0 MAY respond to the request regardless of the Request-URI.

This behavior is common with HTTP/1.1. This behavior can be used as a "traceroute" functionality to check the capabilities of individual hop servers by sending a series of OPTIONS requests with incremented Max-Forwards values.

As is the case for general UA behavior, the transaction layer can return a timeout error if the OPTIONS yields no response. This may indicate that the target is unreachable and hence unavailable.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

An OPTIONS request MAY be sent as part of an established dialog to query the peer on capabilities that may be utilized later in the dialog.

### 11.1 Construction of OPTIONS Request

An OPTIONS request is constructed using the standard rules for a SIP request as discussed in Section 8.1.1.

A Contact header field MAY be present in an OPTIONS.

An Accept header field SHOULD be included to indicate the type of message body the UAC wishes to receive in the response. Typically, this is set to a format that is used to describe the media capabilities of a UA, such as SDP (application/sdp).

The response to an OPTIONS request is assumed to be scoped to the Request-URI in the original request. However, only when an OPTIONS is sent as part of an established dialog is it guaranteed that future requests will be received by the server that generated the OPTIONS response.

Example OPTIONS request:

```
OPTIONS sip:carol@chicago.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKhjhs8ass877
Max-Forwards: 70
To: <sip:carol@chicago.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 63104 OPTIONS
Contact: <sip:alice@pc33.atlanta.com>
Accept: application/sdp
Content-Length: 0
```



### 11.2 Processing of OPTIONS Request

The response to an OPTIONS is constructed using the standard rules for a SIP response as discussed in Section 8.2.6. The response code chosen MUST be the same that would have been chosen had the request been an INVITE. That is, a 200 (OK) would be returned if the UAS is ready to accept a call, a 486 (Busy Here) would be returned if the UAS is busy, etc. This allows an OPTIONS request to be used to determine the basic state of a UAS, which can be an indication of whether the UAS will accept an INVITE request.

An OPTIONS request received within a dialog generates a 200 (OK) response that is identical to one constructed outside a dialog and does not have any impact on the dialog.

This use of OPTIONS has limitations due to the differences in proxy handling of OPTIONS and INVITE requests. While a forked INVITE can result in multiple 200 (OK) responses being returned, a forked OPTIONS will only result in a single 200 (OK) response, since it is treated by proxies using the non-INVITE handling. See Section 16.7

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 700
Keysight, EX-1045
Page 700

for the normative details.

If the response to an OPTIONS is generated by a proxy server, the proxy returns a 200 (OK), listing the capabilities of the server. The response does not contain a message body.

Allow, Accept, Accept-Encoding, Accept-Language, and Supported header fields SHOULD be present in a 200 (OK) response to an OPTIONS request. If the response is generated by a proxy, the Allow header field SHOULD be omitted as it is ambiguous since a proxy is method agnostic. Contact header fields MAY be present in a 200 (OK) response and have the same semantics as in a 3xx response. That is, they may list a set of alternative names and methods of reaching the user. A Warning header field MAY be present.

A message body MAY be sent, the type of which is determined by the Accept header field in the OPTIONS request (application/sdp is the default if the Accept header field is not present). If the types include one that can describe media capabilities, the UAS SHOULD include a body in the response for that purpose. Details on the construction of such a body in the case of application/sdp are described in [13].





Example OPTIONS response generated by a UAS (corresponding to the request in Section 11.1):

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKhjhs8ass877
 ;received=192.0.2.4
To: <sip:carol@chicago.com>;tag=93810874
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 63104 OPTIONS
Contact: <sip:carol@chicago.com>
Contact: <mailto:carol@chicago.com>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE
Accept: application/sdp
Accept-Encoding: gzip
Accept-Language: en
Supported: foo
Content-Type: application/sdp
Content-Length: 274

(SDP not shown)
```

## 12 Dialogs

A key concept for a user agent is that of a dialog. A dialog represents a peer-to-peer SIP relationship between two user agents that persists for some time. The dialog facilitates sequencing of

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 701
Keysight, EX-1045
Page 701

messages between the user agents and proper routing of requests between both of them. The dialog represents a context in which to interpret SIP messages. Section 8 discussed method independent UA processing for requests and responses outside of a dialog. This section discusses how those requests and responses are used to construct a dialog, and then how subsequent requests and responses are sent within a dialog.

A dialog is identified at each UA with a dialog ID, which consists of a Call-ID value, a local tag and a remote tag. The dialog ID at each UA involved in the dialog is not the same. Specifically, the local tag at one UA is identical to the remote tag at the peer UA. The tags are opaque tokens that facilitate the generation of unique dialog IDs.

A dialog ID is also associated with all responses and with any request that contains a tag in the To field. The rules for computing the dialog ID of a message depend on whether the SIP element is a UAC or UAS. For a UAC, the Call-ID value of the dialog ID is set to the Call-ID of the message, the remote tag is set to the tag in the To field of the message, and the local tag is set to the tag in the From





field of the message (these rules apply to both requests and responses). As one would expect for a UAS, the Call-ID value of the dialog ID is set to the Call-ID of the message, the remote tag is set to the tag in the From field of the message, and the local tag is set to the tag in the To field of the message.

A dialog contains certain pieces of state needed for further message transmissions within the dialog. This state consists of the dialog ID, a local sequence number (used to order requests from the UA to its peer), a remote sequence number (used to order requests from its peer to the UA), a local URI, a remote URI, remote target, a boolean flag called "secure", and a route set, which is an ordered list of URIs. The route set is the list of servers that need to be traversed to send a request to the peer. A dialog can also be in the "early" state, which occurs when it is created with a provisional response, and then transition to the "confirmed" state when a 2xx final response arrives. For other responses, or if no response arrives at all on that dialog, the early dialog terminates.

### 12.1 Creation of a Dialog

Dialogs are created through the generation of non-failure responses to requests with specific methods. Within this specification, only 2xx and 101-199 responses with a To tag, where the request was INVITE, will establish a dialog. A dialog established by a non-final response to a request is in the "early" state and it is called an early dialog. Extensions MAY define other means for creating dialogs. Section 13 gives more details that are specific to the INVITE method. Here, we describe the process for creation of dialog state that is not dependent on the method.

UAs MUST assign values to the dialog ID components as described below.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 702
Keysight, EX-1045
Page 702

12.1.1 UAS behavior

When a UAS responds to a request with a response that establishes a dialog (such as a 2xx to INVITE), the UAS MUST copy all Record-Route header field values from the request into the response (including the URIs, URI parameters, and any Record-Route header field parameters, whether they are known or unknown to the UAS) and MUST maintain the order of those values. The UAS MUST add a Contact header field to the response. The Contact header field contains an address where the UAS would like to be contacted for subsequent requests in the dialog (which includes the ACK for a 2xx response in the case of an INVITE). Generally, the host portion of this URI is the IP address or FQDN of the host. The URI provided in the Contact header field MUST be a SIP or SIPS URI. If the request that initiated the dialog contained a





SIPS URI in the Request-URI or in the top Record-Route header field value, if there was any, or the Contact header field if there was no Record-Route header field, the Contact header field in the response MUST be a SIPS URI. The URI SHOULD have global scope (that is, the same URI can be used in messages outside this dialog). The same way, the scope of the URI in the Contact header field of the INVITE is not limited to this dialog either. It can therefore be used in messages to the UAC even outside this dialog.

The UAS then constructs the state of the dialog. This state MUST be maintained for the duration of the dialog.

If the request arrived over TLS, and the Request-URI contained a SIPS URI, the "secure" flag is set to TRUE.

The route set MUST be set to the list of URIs in the Record-Route header field from the request, taken in order and preserving all URI parameters. If no Record-Route header field is present in the request, the route set MUST be set to the empty set. This route set, even if empty, overrides any pre-existing route set for future requests in this dialog. The remote target MUST be set to the URI from the Contact header field of the request.

The remote sequence number MUST be set to the value of the sequence number in the CSeq header field of the request. The local sequence number MUST be empty. The call identifier component of the dialog ID MUST be set to the value of the Call-ID in the request. The local tag component of the dialog ID MUST be set to the tag in the To field in the response to the request (which always includes a tag), and the remote tag component of the dialog ID MUST be set to the tag from the From field in the request. A UAS MUST be prepared to receive a request without a tag in the From field, in which case the tag is considered to have a value of null.

> This is to maintain backwards compatibility with RFC 2543, which did not mandate From tags.

The remote URI MUST be set to the URI in the From field, and the local URI MUST be set to the URI in the To field.

12.1.2 UAC Behavior

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 703
Keysight, EX-1045
Page 703

When a UAC sends a request that can establish a dialog (such as an INVITE) it MUST provide a SIP or SIPS URI with global scope (i.e., the same SIP URI can be used in messages outside this dialog) in the Contact header field of the request. If the request has a Request-URI or a topmost Route header field value with a SIPS URI, the Contact header field MUST contain a SIPS URI.





When a UAC receives a response that establishes a dialog, it constructs the state of the dialog. This state MUST be maintained for the duration of the dialog.

If the request was sent over TLS, and the Request-URI contained a SIPS URI, the "secure" flag is set to TRUE.

The route set MUST be set to the list of URIs in the Record-Route header field from the response, taken in reverse order and preserving all URI parameters. If no Record-Route header field is present in the response, the route set MUST be set to the empty set. This route set, even if empty, overrides any pre-existing route set for future requests in this dialog. The remote target MUST be set to the URI from the Contact header field of the response.

The local sequence number MUST be set to the value of the sequence number in the CSeq header field of the request. The remote sequence number MUST be empty (it is established when the remote UA sends a request within the dialog). The call identifier component of the dialog ID MUST be set to the value of the Call-ID in the request. The local tag component of the dialog ID MUST be set to the tag in the From field in the request, and the remote tag component of the dialog ID MUST be set to the tag in the To field of the response. A UAC MUST be prepared to receive a response without a tag in the To field, in which case the tag is considered to have a value of null.

> This is to maintain backwards compatibility with RFC 2543, which did not mandate To tags.

The remote URI MUST be set to the URI in the To field, and the local URI MUST be set to the URI in the From field.

### 12.2 Requests within a Dialog

Once a dialog has been established between two UAs, either of them MAY initiate new transactions as needed within the dialog. The UA sending the request will take the UAC role for the transaction. The UA receiving the request will take the UAS role. Note that these may be different roles than the UAs held during the transaction that established the dialog.

Requests within a dialog MAY contain Record-Route and Contact header fields. However, these requests do not cause the dialog's route set to be modified, although they may modify the remote target URI. Specifically, requests that are not target refresh requests do not modify the dialog's remote target URI, and requests that are target refresh requests do. For dialogs that have been established with an

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

INVITE, the only target refresh request defined is re-INVITE (see Section 14). Other extensions may define different target refresh requests for dialogs established in other ways.

Note that an ACK is NOT a target refresh request.

Target refresh requests only update the dialog's remote target URI, and not the route set formed from the Record-Route. Updating the latter would introduce severe backwards compatibility problems with RFC 2543-compliant systems.

12.2.1 UAC Behavior

12.2.1.1 Generating the Request

A request within a dialog is constructed by using many of the components of the state stored as part of the dialog.

The URI in the To field of the request MUST be set to the remote URI from the dialog state. The tag in the To header field of the request MUST be set to the remote tag of the dialog ID. The From URI of the request MUST be set to the local URI from the dialog state. The tag in the From header field of the request MUST be set to the local tag of the dialog ID. If the value of the remote or local tags is null, the tag parameter MUST be omitted from the To or From header fields, respectively.

Usage of the URI from the To and From fields in the original request within subsequent requests is done for backwards compatibility with RFC 2543, which used the URI for dialog identification. In this specification, only the tags are used for dialog identification. It is expected that mandatory reflection of the original To and From URI in mid-dialog requests will be deprecated in a subsequent revision of this specification.

The Call-ID of the request MUST be set to the Call-ID of the dialog. Requests within a dialog MUST contain strictly monotonically increasing and contiguous CSeq sequence numbers (increasing-by-one) in each direction (excepting ACK and CANCEL of course, whose numbers equal the requests being acknowledged or cancelled). Therefore, if the local sequence number is not empty, the value of the local sequence number MUST be incremented by one, and this value MUST be placed into the CSeq header field. If the local sequence number is empty, an initial value MUST be chosen using the guidelines of Section 8.1.1.5. The method field in the CSeq header field value MUST match the method of the request.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 705
Keysight, EX-1045
Page 705

With a length of 32 bits, a client could generate, within a single call, one request a second for about 136 years before needing to wrap around. The initial value of the sequence number is chosen so that subsequent requests within the same call will not wrap around. A non-zero initial value allows clients to use a time-based initial sequence number. A client could, for example, choose the 31 most significant bits of a 32-bit second clock as an initial sequence number.

The UAC uses the remote target and route set to build the Request-URI and Route header field of the request.

If the route set is empty, the UAC MUST place the remote target URI into the Request-URI. The UAC MUST NOT add a Route header field to the request.

If the route set is not empty, and the first URI in the route set contains the lr parameter (see Section 19.1.1), the UAC MUST place the remote target URI into the Request-URI and MUST include a Route header field containing the route set values in order, including all parameters.

If the route set is not empty, and its first URI does not contain the lr parameter, the UAC MUST place the first URI from the route set into the Request-URI, stripping any parameters that are not allowed in a Request-URI. The UAC MUST add a Route header field containing the remainder of the route set values in order, including all parameters. The UAC MUST then place the remote target URI into the Route header field as the last value.

For example, if the remote target is sip:user@remoteua and the route set contains:

<sip:proxy1>,<sip:proxy2>,<sip:proxy3;lr>,<sip:proxy4>

The request will be formed with the following Request-URI and Route header field:

METHOD sip:proxy1
Route: <sip:proxy2>,<sip:proxy3;lr>,<sip:proxy4>,<sip:user@remoteua>

If the first URI of the route set does not contain the lr parameter, the proxy indicated does not understand the routing mechanisms described in this document and will act as specified in RFC 2543, replacing the Request-URI with the first Route header field value it receives while forwarding the message. Placing the Request-URI at the end of the Route header field preserves the



information in that Request-URI across the strict router (it will be returned to the Request-URI when the request reaches a loose-router).

A UAC SHOULD include a Contact header field in any target refresh requests within a dialog, and unless there is a need to change it,

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 706
Keysight, EX-1045
Page 706

the URI SHOULD be the same as used in previous requests within the dialog. If the "secure" flag is true, that URI MUST be a SIPS URI. As discussed in Section 12.2.2, a Contact header field in a target refresh request updates the remote target URI. This allows a UA to provide a new contact address, should its address change during the duration of the dialog.

However, requests that are not target refresh requests do not affect the remote target URI for the dialog.

The rest of the request is formed as described in Section 8.1.1.

Once the request has been constructed, the address of the server is computed and the request is sent, using the same procedures for requests outside of a dialog (Section 8.1.2).

> The procedures in Section 8.1.2 will normally result in the request being sent to the address indicated by the topmost Route header field value or the Request-URI if no Route header field is present. Subject to certain restrictions, they allow the request to be sent to an alternate address (such as a default outbound proxy not represented in the route set).

#### 12.2.1.2 Processing the Responses

The UAC will receive responses to the request from the transaction layer. If the client transaction returns a timeout, this is treated as a 408 (Request Timeout) response.

The behavior of a UAC that receives a 3xx response for a request sent within a dialog is the same as if the request had been sent outside a dialog. This behavior is described in Section 8.1.3.4.

> Note, however, that when the UAC tries alternative locations, it still uses the route set for the dialog to build the Route header of the request.

When a UAC receives a 2xx response to a target refresh request, it MUST replace the dialog's remote target URI with the URI from the Contact header field in that response, if present.





If the response for a request within a dialog is a 481 (Call/Transaction Does Not Exist) or a 408 (Request Timeout), the UAC SHOULD terminate the dialog. A UAC SHOULD also terminate a dialog if no response at all is received for the request (the client transaction would inform the TU about the timeout.)

> For INVITE initiated dialogs, terminating the dialog consists of sending a BYE.

#### 12.2.2 UAS Behavior

Requests sent within a dialog, as any other requests, are atomic. If a particular request is accepted by the UAS, all the state changes

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 707
Keysight, EX-1045
Page 707

associated with it are performed. If the request is rejected, none of the state changes are performed.

Note that some requests, such as INVITEs, affect several pieces of state.

The UAS will receive the request from the transaction layer. If the request has a tag in the To header field, the UAS core computes the dialog identifier corresponding to the request and compares it with existing dialogs. If there is a match, this is a mid-dialog request. In that case, the UAS first applies the same processing rules for requests outside of a dialog, discussed in Section 8.2.

If the request has a tag in the To header field, but the dialog identifier does not match any existing dialogs, the UAS may have crashed and restarted, or it may have received a request for a different (possibly failed) UAS (the UASs can construct the To tags so that a UAS can identify that the tag was for a UAS for which it is providing recovery). Another possibility is that the incoming request has been simply misrouted. Based on the To tag, the UAS MAY either accept or reject the request. Accepting the request for acceptable To tags provides robustness, so that dialogs can persist even through crashes. UAs wishing to support this capability must take into consideration some issues such as choosing monotonically increasing CSeq sequence numbers even across reboots, reconstructing the route set, and accepting out-of-range RTP timestamps and sequence numbers.

If the UAS wishes to reject the request because it does not wish to recreate the dialog, it MUST respond to the request with a 481 (Call/Transaction Does Not Exist) status code and pass that to the server transaction.





Requests that do not change in any way the state of a dialog may be received within a dialog (for example, an OPTIONS request). They are processed as if they had been received outside the dialog.

If the remote sequence number is empty, it MUST be set to the value of the sequence number in the CSeq header field value in the request. If the remote sequence number was not empty, but the sequence number of the request is lower than the remote sequence number, the request is out of order and MUST be rejected with a 500 (Server Internal Error) response. If the remote sequence number was not empty, and the sequence number of the request is greater than the remote sequence number, the request is in order. It is possible for the CSeq sequence number to be higher than the remote sequence number by more than one. This is not an error condition, and a UAS SHOULD be prepared to receive and process requests with CSeq values more than one higher than the previous received request. The UAS MUST then set the remote sequence number to the value of the sequence number in the CSeq header field value in the request.

If a proxy challenges a request generated by the UAC, the UAC has

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 708
Keysight, EX-1045
Page 708

to resubmit the request with credentials. The resubmitted request will have a new CSeq number. The UAS will never see the first request, and thus, it will notice a gap in the CSeq number space. Such a gap does not represent any error condition.

When a UAS receives a target refresh request, it MUST replace the dialog's remote target URI with the URI from the Contact header field in that request, if present.

12.3 Termination of a Dialog

Independent of the method, if a request outside of a dialog generates a non-2xx final response, any early dialogs created through provisional responses to that request are terminated. The mechanism for terminating confirmed dialogs is method specific. In this specification, the BYE method terminates a session and the dialog associated with it. See Section 15 for details.

13 Initiating a Session

13.1 Overview

When a user agent client desires to initiate a session (for example, audio, video, or a game), it formulates an INVITE request. The INVITE request asks a server to establish a session. This request may be forwarded by proxies, eventually arriving at one or more UAS that can potentially accept the invitation. These UASs will frequently need to query the user about whether to accept the





invitation. After some time, those UASs can accept the invitation (meaning the session is to be established) by sending a 2xx response. If the invitation is not accepted, a 3xx, 4xx, 5xx or 6xx response is sent, depending on the reason for the rejection. Before sending a final response, the UAS can also send provisional responses (1xx) to advise the UAC of progress in contacting the called user.

After possibly receiving one or more provisional responses, the UAC will get one or more 2xx responses or one non-2xx final response. Because of the protracted amount of time it can take to receive final responses to INVITE, the reliability mechanisms for INVITE transactions differ from those of other requests (like OPTIONS). Once it receives a final response, the UAC needs to send an ACK for every final response it receives. The procedure for sending this ACK depends on the type of response. For final responses between 300 and 699, the ACK processing is done in the transaction layer and follows one set of rules (See Section 17). For 2xx responses, the ACK is generated by the UAC core.

A 2xx response to an INVITE establishes a session, and it also creates a dialog between the UA that issued the INVITE and the UA that generated the 2xx response. Therefore, when multiple 2xx responses are received from different remote UAs (because the INVITE forked), each 2xx establishes a different dialog. All these dialogs are part of the same call.

This section provides details on the establishment of a session using

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 709
Keysight, EX-1045
Page 709

INVITE. A UA that supports INVITE MUST also support ACK, CANCEL and BYE.

13.2 UAC Processing

13.2.1 Creating the Initial INVITE

Since the initial INVITE represents a request outside of a dialog, its construction follows the procedures of Section 8.1.1. Additional processing is required for the specific case of INVITE.

An Allow header field (Section 20.5) SHOULD be present in the INVITE. It indicates what methods can be invoked within a dialog, on the UA sending the INVITE, for the duration of the dialog. For example, a UA capable of receiving INFO requests within a dialog [34] SHOULD include an Allow header field listing the INFO method.

A Supported header field (Section 20.37) SHOULD be present in the INVITE. It enumerates all the extensions understood by the UAC.





An Accept (Section 20.1) header field MAY be present in the INVITE. It indicates which Content-Types are acceptable to the UA, in both the response received by it, and in any subsequent requests sent to it within dialogs established by the INVITE. The Accept header field is especially useful for indicating support of various session description formats.

The UAC MAY add an Expires header field (Section 20.19) to limit the validity of the invitation. If the time indicated in the Expires header field is reached and no final answer for the INVITE has been received, the UAC core SHOULD generate a CANCEL request for the INVITE, as per Section 9.

A UAC MAY also find it useful to add, among others, Subject (Section 20.36), Organization (Section 20.25) and User-Agent (Section 20.41) header fields. They all contain information related to the INVITE.

The UAC MAY choose to add a message body to the INVITE. Section 8.1.1.10 deals with how to construct the header fields -- Content-Type among others -- needed to describe the message body.

There are special rules for message bodies that contain a session description - their corresponding Content-Disposition is "session". SIP uses an offer/answer model where one UA sends a session description, called the offer, which contains a proposed description of the session. The offer indicates the desired communications means (audio, video, games), parameters of those means (such as codec types) and addresses for receiving media from the answerer. The other UA responds with another session description, called the answer, which indicates which communications means are accepted, the parameters that apply to those means, and addresses for receiving media from the offerer. An offer/answer exchange is within the context of a dialog, so that if a SIP INVITE results in multiple dialogs, each is a separate offer/answer exchange. The offer/answer

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 710
Keysight, EX-1045
Page 710

model defines restrictions on when offers and answers can be made (for example, you cannot make a new offer while one is in progress). This results in restrictions on where the offers and answers can appear in SIP messages. In this specification, offers and answers can only appear in INVITE requests and responses, and ACK. The usage of offers and answers is further restricted. For the initial INVITE transaction, the rules are:

- The initial offer MUST be in either an INVITE or, if not there, in the first reliable non-failure message from the UAS back to the UAC. In this specification, that is the final 2xx response.





- If the initial offer is in an INVITE, the answer MUST be in a reliable non-failure message from UAS back to UAC which is correlated to that INVITE. For this specification, that is only the final 2xx response to that INVITE. That same exact answer MAY also be placed in any provisional responses sent prior to the answer. The UAC MUST treat the first session description it receives as the answer, and MUST ignore any session descriptions in subsequent responses to the initial INVITE.

- If the initial offer is in the first reliable non-failure message from the UAS back to UAC, the answer MUST be in the acknowledgement for that message (in this specification, ACK for a 2xx response).

- After having sent or received an answer to the first offer, the UAC MAY generate subsequent offers in requests based on rules specified for that method, but only if it has received answers to any previous offers, and has not sent any offers to which it hasn't gotten an answer.

- Once the UAS has sent or received an answer to the initial offer, it MUST NOT generate subsequent offers in any responses to the initial INVITE. This means that a UAS based on this specification alone can never generate subsequent offers until completion of the initial transaction.

Concretely, the above rules specify two exchanges for UAs compliant to this specification alone - the offer is in the INVITE, and the answer in the 2xx (and possibly in a 1xx as well, with the same value), or the offer is in the 2xx, and the answer is in the ACK. All user agents that support INVITE MUST support these two exchanges.

The Session Description Protocol (SDP) (RFC 2327 [1]) MUST be supported by all user agents as a means to describe sessions, and its usage for constructing offers and answers MUST follow the procedures defined in [13].

The restrictions of the offer-answer model just described only apply to bodies whose Content-Disposition header field value is "session". Therefore, it is possible that both the INVITE and the ACK contain a

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 711
Keysight, EX-1045
Page 711

body message (for example, the INVITE carries a photo (Content-Disposition: render) and the ACK a session description (Content-Disposition: session)).

If the Content-Disposition header field is missing, bodies of Content-Type application/sdp imply the disposition "session", while other content types imply "render".





Once the INVITE has been created, the UAC follows the procedures defined for sending requests outside of a dialog (Section 8). This results in the construction of a client transaction that will ultimately send the request and deliver responses to the UAC.

### 13.2.2 Processing INVITE Responses

Once the INVITE has been passed to the INVITE client transaction, the UAC waits for responses for the INVITE. If the INVITE client transaction returns a timeout rather than a response the TU acts as if a 408 (Request Timeout) response had been received, as described in Section 8.1.3.

#### 13.2.2.1 1xx Responses

Zero, one or multiple provisional responses may arrive before one or more final responses are received. Provisional responses for an INVITE request can create "early dialogs". If a provisional response has a tag in the To field, and if the dialog ID of the response does not match an existing dialog, one is constructed using the procedures defined in Section 12.1.2.

The early dialog will only be needed if the UAC needs to send a request to its peer within the dialog before the initial INVITE transaction completes. Header fields present in a provisional response are applicable as long as the dialog is in the early state (for example, an Allow header field in a provisional response contains the methods that can be used in the dialog while this is in the early state).

#### 13.2.2.2 3xx Responses

A 3xx response may contain one or more Contact header field values providing new addresses where the callee might be reachable. Depending on the status code of the 3xx response (see Section 21.3), the UAC MAY choose to try those new addresses.

#### 13.2.2.3 4xx, 5xx and 6xx Responses

A single non-2xx final response may be received for the INVITE. 4xx, 5xx and 6xx responses may contain a Contact header field value indicating the location where additional information about the error can be found. Subsequent final responses (which would only arrive under error conditions) MUST be ignored.

All early dialogs are considered terminated upon reception of the non-2xx final response.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 712
Keysight, EX-1045
Page 712

After having received the non-2xx final response the UAC core considers the INVITE transaction completed. The INVITE client transaction handles the generation of ACKs for the response (see Section 17).

13.2.2.4 2xx Responses

Multiple 2xx responses may arrive at the UAC for a single INVITE request due to a forking proxy. Each response is distinguished by the tag parameter in the To header field, and each represents a distinct dialog, with a distinct dialog identifier.

If the dialog identifier in the 2xx response matches the dialog identifier of an existing dialog, the dialog MUST be transitioned to the "confirmed" state, and the route set for the dialog MUST be recomputed based on the 2xx response using the procedures of Section 12.2.1.2. Otherwise, a new dialog in the "confirmed" state MUST be constructed using the procedures of Section 12.1.2.

Note that the only piece of state that is recomputed is the route set. Other pieces of state such as the highest sequence numbers (remote and local) sent within the dialog are not recomputed. The route set only is recomputed for backwards compatibility. RFC 2543 did not mandate mirroring of the Record-Route header field in a 1xx, only 2xx. However, we cannot update the entire state of the dialog, since mid-dialog requests may have been sent within the early dialog, modifying the sequence numbers, for example.

The UAC core MUST generate an ACK request for each 2xx received from the transaction layer. The header fields of the ACK are constructed in the same way as for any request sent within a dialog (see Section 12) with the exception of the CSeq and the header fields related to authentication. The sequence number of the CSeq header field MUST be the same as the INVITE being acknowledged, but the CSeq method MUST be ACK. The ACK MUST contain the same credentials as the INVITE. If the 2xx contains an offer (based on the rules above), the ACK MUST carry an answer in its body. If the offer in the 2xx response is not acceptable, the UAC core MUST generate a valid answer in the ACK and then send a BYE immediately.

Once the ACK has been constructed, the procedures of [4] are used to determine the destination address, port and transport. However, the request is passed to the transport layer directly for transmission, rather than a client transaction. This is because the UAC core handles retransmissions of the ACK, not the transaction layer. The ACK MUST be passed to the client transport every time a retransmission of the 2xx final response that triggered the ACK arrives.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 713
Keysight, EX-1045
Page 713

The UAC core considers the INVITE transaction completed 64*T1 seconds after the reception of the first 2xx response. At this point all the early dialogs that have not transitioned to established dialogs are terminated. Once the INVITE transaction is considered completed by the UAC core, no more new 2xx responses are expected to arrive.

If, after acknowledging any 2xx response to an INVITE, the UAC does not want to continue with that dialog, then the UAC MUST terminate the dialog by sending a BYE request as described in Section 15.

13.3 UAS Processing

13.3.1 Processing of the INVITE

The UAS core will receive INVITE requests from the transaction layer. It first performs the request processing procedures of Section 8.2, which are applied for both requests inside and outside of a dialog.

Assuming these processing states are completed without generating a response, the UAS core performs the additional processing steps:

1. If the request is an INVITE that contains an Expires header field, the UAS core sets a timer for the number of seconds indicated in the header field value. When the timer fires, the invitation is considered to be expired. If the invitation expires before the UAS has generated a final response, a 487 (Request Terminated) response SHOULD be generated.

2. If the request is a mid-dialog request, the method-independent processing described in Section 12.2.2 is first applied. It might also modify the session; Section 14 provides details.

3. If the request has a tag in the To header field but the dialog identifier does not match any of the existing dialogs, the UAS may have crashed and restarted, or may have received a request for a different (possibly failed) UAS. Section 12.2.2 provides guidelines to achieve a robust behavior under such a situation.

Processing from here forward assumes that the INVITE is outside of a dialog, and is thus for the purposes of establishing a new session.

The INVITE may contain a session description, in which case the UAS is being presented with an offer for that session. It is possible that the user is already a participant in that session, even though the INVITE is outside of a dialog. This can happen when a user is invited to the same multicast conference by multiple other participants. If desired, the UAS MAY use identifiers within the session description to detect this duplication. For example, SDP



contains a session id and version number in the origin (o) field. If the user is already a member of the session, and the session parameters contained in the session description have not changed, the UAS MAY silently accept the INVITE (that is, send a 2xx response without prompting the user).

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 714
Keysight, EX-1045
Page 714

If the INVITE does not contain a session description, the UAS is being asked to participate in a session, and the UAC has asked that the UAS provide the offer of the session. It MUST provide the offer in its first non-failure reliable message back to the UAC. In this specification, that is a 2xx response to the INVITE.

The UAS can indicate progress, accept, redirect, or reject the invitation. In all of these cases, it formulates a response using the procedures described in Section 8.2.6.

13.3.1.1 Progress

If the UAS is not able to answer the invitation immediately, it can choose to indicate some kind of progress to the UAC (for example, an indication that a phone is ringing). This is accomplished with a provisional response between 101 and 199. These provisional responses establish early dialogs and therefore follow the procedures of Section 12.1.1 in addition to those of Section 8.2.6. A UAS MAY send as many provisional responses as it likes. Each of these MUST indicate the same dialog ID. However, these will not be delivered reliably.

If the UAS desires an extended period of time to answer the INVITE, it will need to ask for an "extension" in order to prevent proxies from canceling the transaction. A proxy has the option of canceling a transaction when there is a gap of 3 minutes between responses in a transaction. To prevent cancellation, the UAS MUST send a non-100 provisional response at every minute, to handle the possibility of lost provisional responses.

An INVITE transaction can go on for extended durations when the user is placed on hold, or when interworking with PSTN systems which allow communications to take place without answering the call. The latter is common in Interactive Voice Response (IVR) systems.

13.3.1.2 The INVITE is Redirected

If the UAS decides to redirect the call, a 3xx response is sent. A 300 (Multiple Choices), 301 (Moved Permanently) or 302 (Moved Temporarily) response SHOULD contain a Contact header field





containing one or more URIs of new addresses to be tried. The response is passed to the INVITE server transaction, which will deal with its retransmissions.

13.3.1.3 The INVITE is Rejected

A common scenario occurs when the callee is currently not willing or able to take additional calls at this end system. A 486 (Busy Here) SHOULD be returned in such a scenario. If the UAS knows that no other end system will be able to accept this call, a 600 (Busy Everywhere) response SHOULD be sent instead. However, it is unlikely that a UAS will be able to know this in general, and thus this response will not usually be used. The response is passed to the

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

INVITE server transaction, which will deal with its retransmissions.

A UAS rejecting an offer contained in an INVITE SHOULD return a 488 (Not Acceptable Here) response. Such a response SHOULD include a Warning header field value explaining why the offer was rejected.

13.3.1.4 The INVITE is Accepted

The UAS core generates a 2xx response. This response establishes a dialog, and therefore follows the procedures of Section 12.1.1 in addition to those of Section 8.2.6.

A 2xx response to an INVITE SHOULD contain the Allow header field and the Supported header field, and MAY contain the Accept header field. Including these header fields allows the UAC to determine the features and extensions supported by the UAS for the duration of the call, without probing.

If the INVITE request contained an offer, and the UAS had not yet sent an answer, the 2xx MUST contain an answer. If the INVITE did not contain an offer, the 2xx MUST contain an offer if the UAS had not yet sent an offer.

Once the response has been constructed, it is passed to the INVITE server transaction. Note, however, that the INVITE server transaction will be destroyed as soon as it receives this final response and passes it to the transport. Therefore, it is necessary to periodically pass the response directly to the transport until the ACK arrives. The 2xx response is passed to the transport with an interval that starts at T1 seconds and doubles for each retransmission until it reaches T2 seconds (T1 and T2 are defined in Section 17). Response retransmissions cease when an ACK request for the response is received. This is independent of whatever transport protocols are used to send the response.





Since 2xx is retransmitted end-to-end, there may be hops between UAS and UAC that are UDP. To ensure reliable delivery across these hops, the response is retransmitted periodically even if the transport at the UAS is reliable.

If the server retransmits the 2xx response for 64*T1 seconds without receiving an ACK, the dialog is confirmed, but the session SHOULD be terminated. This is accomplished with a BYE, as described in Section 15.

14 Modifying an Existing Session

A successful INVITE request (see Section 13) establishes both a dialog between two user agents and a session using the offer-answer model. Section 12 explains how to modify an existing dialog using a target refresh request (for example, changing the remote target URI of the dialog). This section describes how to modify the actual session. This modification can involve changing addresses or ports, adding a media stream, deleting a media stream, and so on. This is accomplished by sending a new INVITE request within the same dialog

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 716
Keysight, EX-1045
Page 716

that established the session. An INVITE request sent within an existing dialog is known as a re-INVITE.

Note that a single re-INVITE can modify the dialog and the parameters of the session at the same time.

Either the caller or callee can modify an existing session.

The behavior of a UA on detection of media failure is a matter of local policy. However, automated generation of re-INVITE or BYE is NOT RECOMMENDED to avoid flooding the network with traffic when there is congestion. In any case, if these messages are sent automatically, they SHOULD be sent after some randomized interval.

Note that the paragraph above refers to automatically generated BYEs and re-INVITEs. If the user hangs up upon media failure, the UA would send a BYE request as usual.

14.1 UAC Behavior

The same offer-answer model that applies to session descriptions in INVITEs (Section 13.2.1) applies to re-INVITEs. As a result, a UAC that wants to add a media stream, for example, will create a new offer that contains this media stream, and send that in an INVITE request to its peer. It is important to note that the full description of the session, not just the change, is sent. This supports stateless session processing in various elements, and supports failover and recovery capabilities. Of course, a UAC MAY





send a re-INVITE with no session description, in which case the first reliable non-failure response to the re-INVITE will contain the offer (in this specification, that is a 2xx response).

If the session description format has the capability for version numbers, the offerer SHOULD indicate that the version of the session description has changed.

The To, From, Call-ID, CSeq, and Request-URI of a re-INVITE are set following the same rules as for regular requests within an existing dialog, described in Section 12.

A UAC MAY choose not to add an Alert-Info header field or a body with Content-Disposition "alert" to re-INVITEs because UASs do not typically alert the user upon reception of a re-INVITE.

Unlike an INVITE, which can fork, a re-INVITE will never fork, and therefore, only ever generate a single final response. The reason a re-INVITE will never fork is that the Request-URI identifies the target as the UA instance it established the dialog with, rather than identifying an address-of-record for the user.

Note that a UAC MUST NOT initiate a new INVITE transaction within a dialog while another INVITE transaction is in progress in either direction.

1. If there is an ongoing INVITE client transaction, the TU MUST

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

wait until the transaction reaches the completed or terminated state before initiating the new INVITE.

2. If there is an ongoing INVITE server transaction, the TU MUST wait until the transaction reaches the confirmed or terminated state before initiating the new INVITE.

However, a UA MAY initiate a regular transaction while an INVITE transaction is in progress. A UA MAY also initiate an INVITE transaction while a regular transaction is in progress.

If a UA receives a non-2xx final response to a re-INVITE, the session parameters MUST remain unchanged, as if no re-INVITE had been issued. Note that, as stated in Section 12.2.1.2, if the non-2xx final response is a 481 (Call/Transaction Does Not Exist), or a 408 (Request Timeout), or no response at all is received for the re-INVITE (that is, a timeout is returned by the INVITE client transaction), the UAC will terminate the dialog.





If a UAC receives a 491 response to a re-INVITE, it SHOULD start a timer with a value T chosen as follows:

1. If the UAC is the owner of the Call-ID of the dialog ID (meaning it generated the value), T has a randomly chosen value between 2.1 and 4 seconds in units of 10 ms.

2. If the UAC is not the owner of the Call-ID of the dialog ID, T has a randomly chosen value of between 0 and 2 seconds in units of 10 ms.

When the timer fires, the UAC SHOULD attempt the re-INVITE once more, if it still desires for that session modification to take place. For example, if the call was already hung up with a BYE, the re-INVITE would not take place.

The rules for transmitting a re-INVITE and for generating an ACK for a 2xx response to re-INVITE are the same as for the initial INVITE (Section 13.2.1).

### 14.2 UAS Behavior

Section 13.3.1 describes the procedure for distinguishing incoming re-INVITEs from incoming initial INVITEs and handling a re-INVITE for an existing dialog.

A UAS that receives a second INVITE before it sends the final response to a first INVITE with a lower CSeq sequence number on the same dialog MUST return a 500 (Server Internal Error) response to the second INVITE and MUST include a Retry-After header field with a randomly chosen value of between 0 and 10 seconds.

A UAS that receives an INVITE on a dialog while an INVITE it had sent on that dialog is in progress MUST return a 491 (Request Pending)

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 718
Keysight, EX-1045
Page 718

response to the received INVITE.

If a UA receives a re-INVITE for an existing dialog, it MUST check any version identifiers in the session description or, if there are no version identifiers, the content of the session description to see if it has changed. If the session description has changed, the UAS MUST adjust the session parameters accordingly, possibly after asking the user for confirmation.

> Versioning of the session description can be used to accommodate the capabilities of new arrivals to a conference, add or delete media, or change from a unicast to a multicast conference.





If the new session description is not acceptable, the UAS can reject it by returning a 488 (Not Acceptable Here) response for the re-INVITE. This response SHOULD include a Warning header field.

If a UAS generates a 2xx response and never receives an ACK, it SHOULD generate a BYE to terminate the dialog.

A UAS MAY choose not to generate 180 (Ringing) responses for a re-INVITE because UACs do not typically render this information to the user. For the same reason, UASs MAY choose not to use an Alert-Info header field or a body with Content-Disposition "alert" in responses to a re-INVITE.

A UAS providing an offer in a 2xx (because the INVITE did not contain an offer) SHOULD construct the offer as if the UAS were making a brand new call, subject to the constraints of sending an offer that updates an existing session, as described in [13] in the case of SDP. Specifically, this means that it SHOULD include as many media formats and media types that the UA is willing to support. The UAS MUST ensure that the session description overlaps with its previous session description in media formats, transports, or other parameters that require support from the peer. This is to avoid the need for the peer to reject the session description. If, however, it is unacceptable to the UAC, the UAC SHOULD generate an answer with a valid session description, and then send a BYE to terminate the session.

## 15 Terminating a Session

This section describes the procedures for terminating a session established by SIP. The state of the session and the state of the dialog are very closely related. When a session is initiated with an INVITE, each 1xx or 2xx response from a distinct UAS creates a dialog, and if that response completes the offer/answer exchange, it also creates a session. As a result, each session is "associated" with a single dialog - the one which resulted in its creation. If an initial INVITE generates a non-2xx final response, that terminates all sessions (if any) and all dialogs (if any) that were created through responses to the request. By virtue of completing the transaction, a non-2xx final response also prevents further sessions from being created as a result of the INVITE. The BYE request is

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 719
Keysight, EX-1045
Page 719

used to terminate a specific session or attempted session. In this case, the specific session is the one with the peer UA on the other side of the dialog. When a BYE is received on a dialog, any session associated with that dialog SHOULD terminate. A UA MUST NOT send a BYE outside of a dialog. The caller's UA MAY send a BYE for either confirmed or early dialogs, and the callee's UA MAY send a BYE on confirmed dialogs, but MUST NOT send a BYE on early dialogs.





However, the callee's UA MUST NOT send a BYE on a confirmed dialog until it has received an ACK for its 2xx response or until the server transaction times out. If no SIP extensions have defined other application layer states associated with the dialog, the BYE also terminates the dialog.

The impact of a non-2xx final response to INVITE on dialogs and sessions makes the use of CANCEL attractive. The CANCEL attempts to force a non-2xx response to the INVITE (in particular, a 487). Therefore, if a UAC wishes to give up on its call attempt entirely, it can send a CANCEL. If the INVITE results in 2xx final response(s) to the INVITE, this means that a UAS accepted the invitation while the CANCEL was in progress. The UAC MAY continue with the sessions established by any 2xx responses, or MAY terminate them with BYE.

> The notion of "hanging up" is not well defined within SIP. It is specific to a particular, albeit common, user interface. Typically, when the user hangs up, it indicates a desire to terminate the attempt to establish a session, and to terminate any sessions already created. For the caller's UA, this would imply a CANCEL request if the initial INVITE has not generated a final response, and a BYE to all confirmed dialogs after a final response. For the callee's UA, it would typically imply a BYE; presumably, when the user picked up the phone, a 2xx was generated, and so hanging up would result in a BYE after the ACK is received. This does not mean a user cannot hang up before receipt of the ACK, it just means that the software in his phone needs to maintain state for a short while in order to clean up properly. If the particular UI allows for the user to reject a call before its answered, a 403 (Forbidden) is a good way to express that. As per the rules above, a BYE can't be sent.

## 15.1 Terminating a Session with a BYE Request

### 15.1.1 UAC Behavior

A BYE request is constructed as would any other request within a dialog, as described in Section 12.

Once the BYE is constructed, the UAC core creates a new non-INVITE client transaction, and passes it the BYE request. The UAC MUST consider the session terminated (and therefore stop sending or listening for media) as soon as the BYE request is passed to the client transaction. If the response for the BYE is a 481 (Call/Transaction Does Not Exist) or a 408 (Request Timeout) or no

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 720
Keysight, EX-1045
Page 720

response at all is received for the BYE (that is, a timeout is returned by the client transaction), the UAC MUST consider the session and the dialog terminated.

### 15.1.2 UAS Behavior

A UAS first processes the BYE request according to the general UAS processing described in Section 8.2. A UAS core receiving a BYE request checks if it matches an existing dialog. If the BYE does not match an existing dialog, the UAS core SHOULD generate a 481 (Call/Transaction Does Not Exist) response and pass that to the server transaction.

> This rule means that a BYE sent without tags by a UAC will be rejected. This is a change from RFC 2543, which allowed BYE without tags.

A UAS core receiving a BYE request for an existing dialog MUST follow the procedures of Section 12.2.2 to process the request. Once done, the UAS SHOULD terminate the session (and therefore stop sending and listening for media). The only case where it can elect not to are multicast sessions, where participation is possible even if the other participant in the dialog has terminated its involvement in the session. Whether or not it ends its participation on the session, the UAS core MUST generate a 2xx response to the BYE, and MUST pass that to the server transaction for transmission.

The UAS MUST still respond to any pending requests received for that dialog. It is RECOMMENDED that a 487 (Request Terminated) response be generated to those pending requests.

## 16 Proxy Behavior

### 16.1 Overview

SIP proxies are elements that route SIP requests to user agent servers and SIP responses to user agent clients. A request may traverse several proxies on its way to a UAS. Each will make routing decisions, modifying the request before forwarding it to the next element. Responses will route through the same set of proxies traversed by the request in the reverse order.

Being a proxy is a logical role for a SIP element. When a request arrives, an element that can play the role of a proxy first decides if it needs to respond to the request on its own. For instance, the request may be malformed or the element may need credentials from the client before acting as a proxy. The element MAY respond with any



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 721
Keysight, EX-1045
Page 721

appropriate error code. When responding directly to a request, the element is playing the role of a UAS and MUST behave as described in Section 8.2.

A proxy can operate in either a stateful or stateless mode for each new request. When stateless, a proxy acts as a simple forwarding element. It forwards each request downstream to a single element determined by making a targeting and routing decision based on the request. It simply forwards every response it receives upstream. A stateless proxy discards information about a message once the message has been forwarded. A stateful proxy remembers information (specifically, transaction state) about each incoming request and any requests it sends as a result of processing the incoming request. It uses this information to affect the processing of future messages associated with that request. A stateful proxy MAY choose to "fork" a request, routing it to multiple destinations. Any request that is forwarded to more than one location MUST be handled statefully.

In some circumstances, a proxy MAY forward requests using stateful transports (such as TCP) without being transaction-stateful. For instance, a proxy MAY forward a request from one TCP connection to another transaction statelessly as long as it places enough information in the message to be able to forward the response down the same connection the request arrived on. Requests forwarded between different types of transports where the proxy's TU must take an active role in ensuring reliable delivery on one of the transports MUST be forwarded transaction statefully.

A stateful proxy MAY transition to stateless operation at any time during the processing of a request, so long as it did not do anything that would otherwise prevent it from being stateless initially (forking, for example, or generation of a 100 response). When performing such a transition, all state is simply discarded. The proxy SHOULD NOT initiate a CANCEL request.

Much of the processing involved when acting statelessly or statefully for a request is identical. The next several subsections are written from the point of view of a stateful proxy. The last section calls out those places where a stateless proxy behaves differently.

### 16.2 Stateful Proxy

When stateful, a proxy is purely a SIP transaction processing engine. Its behavior is modeled here in terms of the server and client transactions defined in Section 17. A stateful proxy has a server transaction associated with one or more client transactions by a higher layer proxy processing component (see figure 3), known as a proxy core. An incoming request is processed by a server



transaction. Requests from the server transaction are passed to a proxy core. The proxy core determines where to route the request, choosing one or more next-hop locations. An outgoing request for each next-hop location is processed by its own associated client transaction. The proxy core collects the responses from the client transactions and uses them to send responses to the server

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 722
Keysight, EX-1045
Page 722

transaction.

A stateful proxy creates a new server transaction for each new request received. Any retransmissions of the request will then be handled by that server transaction per Section 17. The proxy core MUST behave as a UAS with respect to sending an immediate provisional on that server transaction (such as 100 Trying) as described in Section 8.2.6. Thus, a stateful proxy SHOULD NOT generate 100 (Trying) responses to non-INVITE requests.

This is a model of proxy behavior, not of software. An implementation is free to take any approach that replicates the external behavior this model defines.

For all new requests, including any with unknown methods, an element intending to proxy the request MUST:

1. Validate the request (Section 16.3)

2. Preprocess routing information (Section 16.4)

3. Determine target(s) for the request (Section 16.5)

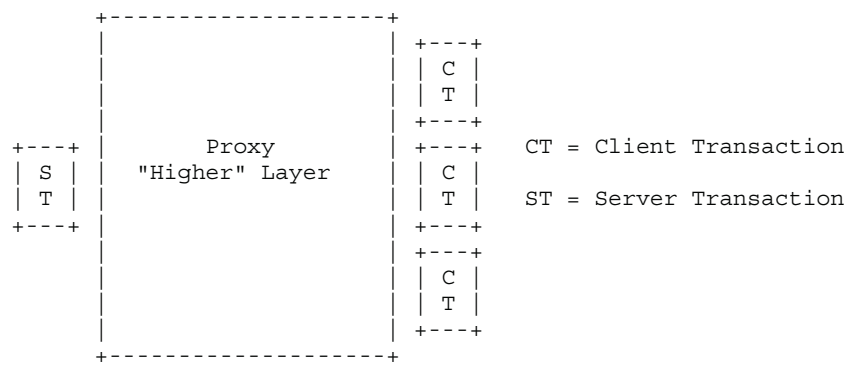



Figure 3: Stateful Proxy Model



4. Forward the request to each target (Section 16.6)

5. Process all responses (Section 16.7)

## 16.3 Request Validation

Before an element can proxy a request, it MUST verify the message's validity. A valid message must pass the following checks:

1. Reasonable Syntax

2. URI scheme

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 723
Keysight, EX-1045
Page 723

3. Max-Forwards

4. (Optional) Loop Detection

5. Proxy-Require

6. Proxy-Authorization

If any of these checks fail, the element MUST behave as a user agent server (see Section 8.2) and respond with an error code.

Notice that a proxy is not required to detect merged requests and MUST NOT treat merged requests as an error condition. The endpoints receiving the requests will resolve the merge as described in Section 8.2.2.2.

1. Reasonable syntax check

The request MUST be well-formed enough to be handled with a server transaction. Any components involved in the remainder of these Request Validation steps or the Request Forwarding section MUST be well-formed. Any other components, well-formed or not, SHOULD be ignored and remain unchanged when the message is forwarded. For instance, an element would not reject a request because of a malformed Date header field. Likewise, a proxy would not remove a malformed Date header field before forwarding a request.

This protocol is designed to be extended. Future extensions may define new methods and header fields at any time. An element MUST NOT refuse to proxy a request because it contains a method or header field it does not know about.





2. URI scheme check

If the Request-URI has a URI whose scheme is not understood by the proxy, the proxy SHOULD reject the request with a 416 (Unsupported URI Scheme) response.

3. Max-Forwards check

The Max-Forwards header field (Section 20.22) is used to limit the number of elements a SIP request can traverse.

If the request does not contain a Max-Forwards header field, this check is passed.

If the request contains a Max-Forwards header field with a field value greater than zero, the check is passed.

If the request contains a Max-Forwards header field with a field value of zero (0), the element MUST NOT forward the request. If the request was for OPTIONS, the element MAY act as the final

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

recipient and respond per Section 11. Otherwise, the element MUST return a 483 (Too many hops) response.

4. Optional Loop Detection check

An element MAY check for forwarding loops before forwarding a request. If the request contains a Via header field with a sent-by value that equals a value placed into previous requests by the proxy, the request has been forwarded by this element before. The request has either looped or is legitimately spiraling through the element. To determine if the request has looped, the element MAY perform the branch parameter calculation described in Step 8 of Section 16.6 on this message and compare it to the parameter received in that Via header field. If the parameters match, the request has looped. If they differ, the request is spiraling, and processing continues. If a loop is detected, the element MAY return a 482 (Loop Detected) response.

5. Proxy-Require check

Future extensions to this protocol may introduce features that require special handling by proxies. Endpoints will include a Proxy-Require header field in requests that use these features, telling the proxy not to process the request unless the feature is understood.





If the request contains a Proxy-Require header field (Section 20.29) with one or more option-tags this element does not understand, the element MUST return a 420 (Bad Extension) response. The response MUST include an Unsupported (Section 20.40) header field listing those option-tags the element did not understand.

6. Proxy-Authorization check

If an element requires credentials before forwarding a request, the request MUST be inspected as described in Section 22.3. That section also defines what the element must do if the inspection fails.

16.4 Route Information Preprocessing

The proxy MUST inspect the Request-URI of the request. If the Request-URI of the request contains a value this proxy previously placed into a Record-Route header field (see Section 16.6 item 4), the proxy MUST replace the Request-URI in the request with the last value from the Route header field, and remove that value from the Route header field. The proxy MUST then proceed as if it received this modified request.

This will only happen when the element sending the request to the proxy (which may have been an endpoint) is a strict router. This rewrite on receive is necessary to enable backwards compatibility

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 725
Keysight, EX-1045
Page 725

with those elements. It also allows elements following this specification to preserve the Request-URI through strict-routing proxies (see Section 12.2.1.1).

This requirement does not obligate a proxy to keep state in order to detect URIs it previously placed in Record-Route header fields. Instead, a proxy need only place enough information in those URIs to recognize them as values it provided when they later appear.

If the Request-URI contains a maddr parameter, the proxy MUST check to see if its value is in the set of addresses or domains the proxy is configured to be responsible for. If the Request-URI has a maddr parameter with a value the proxy is responsible for, and the request was received using the port and transport indicated (explicitly or by default) in the Request-URI, the proxy MUST strip the maddr and any non-default port or transport parameter and continue processing as if those values had not been present in the request.





A request may arrive with a maddr matching the proxy, but on a port or transport different from that indicated in the URI. Such a request needs to be forwarded to the proxy using the indicated port and transport.

If the first value in the Route header field indicates this proxy, the proxy MUST remove that value from the request.

16.5 Determining Request Targets

Next, the proxy calculates the target(s) of the request. The set of targets will either be predetermined by the contents of the request or will be obtained from an abstract location service. Each target in the set is represented as a URI.

If the Request-URI of the request contains an maddr parameter, the Request-URI MUST be placed into the target set as the only target URI, and the proxy MUST proceed to Section 16.6.

If the domain of the Request-URI indicates a domain this element is not responsible for, the Request-URI MUST be placed into the target set as the only target, and the element MUST proceed to the task of Request Forwarding (Section 16.6).

There are many circumstances in which a proxy might receive a request for a domain it is not responsible for. A firewall proxy handling outgoing calls (the way HTTP proxies handle outgoing requests) is an example of where this is likely to occur.

If the target set for the request has not been predetermined as described above, this implies that the element is responsible for the domain in the Request-URI, and the element MAY use whatever mechanism it desires to determine where to send the request. Any of these mechanisms can be modeled as accessing an abstract Location Service.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 726
Keysight, EX-1045
Page 726

This may consist of obtaining information from a location service created by a SIP Registrar, reading a database, consulting a presence server, utilizing other protocols, or simply performing an algorithmic substitution on the Request-URI. When accessing the location service constructed by a registrar, the Request-URI MUST first be canonicalized as described in Section 10.3 before being used as an index. The output of these mechanisms is used to construct the target set.

If the Request-URI does not provide sufficient information for the proxy to determine the target set, it SHOULD return a 485 (Ambiguous) response. This response SHOULD contain a Contact header field containing URIs of new addresses to be tried. For example, an INVITE





to sip:John.Smith@company.com may be ambiguous at a proxy whose location service has multiple John Smiths listed. See Section 21.4.23 for details.

Any information in or about the request or the current environment of the element MAY be used in the construction of the target set. For instance, different sets may be constructed depending on contents or the presence of header fields and bodies, the time of day of the request's arrival, the interface on which the request arrived, failure of previous requests, or even the element's current level of utilization.

As potential targets are located through these services, their URIs are added to the target set. Targets can only be placed in the target set once. If a target URI is already present in the set (based on the definition of equality for the URI type), it MUST NOT be added again.

A proxy MUST NOT add additional targets to the target set if the Request-URI of the original request does not indicate a resource this proxy is responsible for.

> A proxy can only change the Request-URI of a request during forwarding if it is responsible for that URI. If the proxy is not responsible for that URI, it will not recurse on 3xx or 416 responses as described below.

If the Request-URI of the original request indicates a resource this proxy is responsible for, the proxy MAY continue to add targets to the set after beginning Request Forwarding. It MAY use any information obtained during that processing to determine new targets. For instance, a proxy may choose to incorporate contacts obtained in a redirect response (3xx) into the target set. If a proxy uses a dynamic source of information while building the target set (for instance, if it consults a SIP Registrar), it SHOULD monitor that source for the duration of processing the request. New locations SHOULD be added to the target set as they become available. As above, any given URI MUST NOT be added to the set more than once.

> Allowing a URI to be added to the set only once reduces unnecessary network traffic, and in the case of incorporating

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

contacts from redirect requests prevents infinite recursion.

For example, a trivial location service is a "no-op", where the target URI is equal to the incoming request URI. The request is sent to a specific next hop proxy for further processing. During request





forwarding of Section 16.6, Item 6, the identity of that next hop, expressed as a SIP or SIPS URI, is inserted as the top-most Route header field value into the request.

If the Request-URI indicates a resource at this proxy that does not exist, the proxy MUST return a 404 (Not Found) response.

If the target set remains empty after applying all of the above, the proxy MUST return an error response, which SHOULD be the 480 (Temporarily Unavailable) response.

16.6 Request Forwarding

As soon as the target set is non-empty, a proxy MAY begin forwarding the request. A stateful proxy MAY process the set in any order. It MAY process multiple targets serially, allowing each client transaction to complete before starting the next. It MAY start client transactions with every target in parallel. It also MAY arbitrarily divide the set into groups, processing the groups serially and processing the targets in each group in parallel.

A common ordering mechanism is to use the qvalue parameter of targets obtained from Contact header fields (see Section 20.10). Targets are processed from highest qvalue to lowest. Targets with equal qvalues may be processed in parallel.

A stateful proxy must have a mechanism to maintain the target set as responses are received and associate the responses to each forwarded request with the original request. For the purposes of this model, this mechanism is a "response context" created by the proxy layer before forwarding the first request.

For each target, the proxy forwards the request following these steps:

1. Make a copy of the received request

2. Update the Request-URI

3. Update the Max-Forwards header field

4. Optionally add a Record-route header field value

5. Optionally add additional header fields

6. Postprocess routing information

7. Determine the next-hop address, port, and transport

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

8. Add a Via header field value

9. Add a Content-Length header field if necessary

10. Forward the new request

11. Set timer C

Each of these steps is detailed below:

1. Copy request

   The proxy starts with a copy of the received request. The copy MUST initially contain all of the header fields from the received request. Fields not detailed in the processing described below MUST NOT be removed. The copy SHOULD maintain the ordering of the header fields as in the received request. The proxy MUST NOT reorder field values with a common field name (See Section 7.3.1). The proxy MUST NOT add to, modify, or remove the message body.

   An actual implementation need not perform a copy; the primary requirement is that the processing for each next hop begin with the same request.

2. Request-URI

   The Request-URI in the copy's start line MUST be replaced with the URI for this target. If the URI contains any parameters not allowed in a Request-URI, they MUST be removed.

   This is the essence of a proxy's role. This is the mechanism through which a proxy routes a request toward its destination.

   In some circumstances, the received Request-URI is placed into the target set without being modified. For that target, the replacement above is effectively a no-op.

3. Max-Forwards

   If the copy contains a Max-Forwards header field, the proxy MUST decrement its value by one (1).

   If the copy does not contain a Max-Forwards header field, the proxy MUST add one with a field value, which SHOULD be 70.

   Some existing UAs will not provide a Max-Forwards header field in a request.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

4. Record-Route

If this proxy wishes to remain on the path of future requests in a dialog created by this request (assuming the request creates a dialog), it MUST insert a Record-Route header field value into the copy before any existing Record-Route header field values, even if a Route header field is already present.

Requests establishing a dialog may contain a preloaded Route header field.

If this request is already part of a dialog, the proxy SHOULD insert a Record-Route header field value if it wishes to remain on the path of future requests in the dialog. In normal endpoint operation as described in Section 12, these Record-Route header field values will not have any effect on the route sets used by the endpoints.

The proxy will remain on the path if it chooses to not insert a Record-Route header field value into requests that are already part of a dialog. However, it would be removed from the path when an endpoint that has failed reconstitutes the dialog.

A proxy MAY insert a Record-Route header field value into any request. If the request does not initiate a dialog, the endpoints will ignore the value. See Section 12 for details on how endpoints use the Record-Route header field values to construct Route header fields.

Each proxy in the path of a request chooses whether to add a Record-Route header field value independently - the presence of a Record-Route header field in a request does not obligate this proxy to add a value.

The URI placed in the Record-Route header field value MUST be a SIP or SIPS URI. This URI MUST contain an lr parameter (see Section 19.1.1). This URI MAY be different for each destination the request is forwarded to. The URI SHOULD NOT contain the transport parameter unless the proxy has knowledge (such as in a private network) that the next downstream element that will be in the path of subsequent requests supports that transport.

The URI this proxy provides will be used by some other element to make a routing decision. This proxy, in general, has no way of knowing the capabilities of that element, so it must restrict itself to the mandatory elements of a SIP implementation: SIP URIs and either the TCP or UDP transports.



The URI placed in the Record-Route header field MUST resolve to the element inserting it (or a suitable stand-in) when the server location procedures of [4] are applied to it, so that subsequent requests reach the same SIP element. If the Request-URI contains a SIPS URI, or the topmost Route header field value (after the post processing of bullet 6) contains a

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 730
Keysight, EX-1045
Page 730

SIPS URI, the URI placed into the Record-Route header field MUST be a SIPS URI. Furthermore, if the request was not received over TLS, the proxy MUST insert a Record-Route header field. In a similar fashion, a proxy that receives a request over TLS, but generates a request without a SIPS URI in the Request-URI or topmost Route header field value (after the post processing of bullet 6), MUST insert a Record-Route header field that is not a SIPS URI.

A proxy at a security perimeter must remain on the perimeter throughout the dialog.

If the URI placed in the Record-Route header field needs to be rewritten when it passes back through in a response, the URI MUST be distinct enough to locate at that time. (The request may spiral through this proxy, resulting in more than one Record-Route header field value being added). Item 8 of Section 16.7 recommends a mechanism to make the URI sufficiently distinct.

The proxy MAY include parameters in the Record-Route header field value. These will be echoed in some responses to the request such as the 200 (OK) responses to INVITE. Such parameters may be useful for keeping state in the message rather than the proxy.

If a proxy needs to be in the path of any type of dialog (such as one straddling a firewall), it SHOULD add a Record-Route header field value to every request with a method it does not understand since that method may have dialog semantics.

The URI a proxy places into a Record-Route header field is only valid for the lifetime of any dialog created by the transaction in which it occurs. A dialog-stateful proxy, for example, MAY refuse to accept future requests with that value in the Request-URI after the dialog has terminated. Non-dialog-stateful proxies, of course, have no concept of when the dialog has terminated, but they MAY encode enough information in the value to compare it against the dialog identifier of future requests and MAY reject requests not matching that information. Endpoints MUST NOT use a URI obtained from a Record-Route header field outside the dialog in which it was provided. See





Section 12 for more information on an endpoint's use of Record-Route header fields.

Record-routing may be required by certain services where the proxy needs to observe all messages in a dialog. However, it slows down processing and impairs scalability and thus proxies should only record-route if required for a particular service.

The Record-Route process is designed to work for any SIP request that initiates a dialog. INVITE is the only such request in this specification, but extensions to the protocol MAY define others.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 731
Keysight, EX-1045
Page 731

5. Add Additional Header Fields

   The proxy MAY add any other appropriate header fields to the copy at this point.

6. Postprocess routing information

   A proxy MAY have a local policy that mandates that a request visit a specific set of proxies before being delivered to the destination. A proxy MUST ensure that all such proxies are loose routers. Generally, this can only be known with certainty if the proxies are within the same administrative domain. This set of proxies is represented by a set of URIs (each of which contains the lr parameter). This set MUST be pushed into the Route header field of the copy ahead of any existing values, if present. If the Route header field is absent, it MUST be added, containing that list of URIs.

   If the proxy has a local policy that mandates that the request visit one specific proxy, an alternative to pushing a Route value into the Route header field is to bypass the forwarding logic of item 10 below, and instead just send the request to the address, port, and transport for that specific proxy. If the request has a Route header field, this alternative MUST NOT be used unless it is known that next hop proxy is a loose router. Otherwise, this approach MAY be used, but the Route insertion mechanism above is preferred for its robustness, flexibility, generality and consistency of operation. Furthermore, if the Request-URI contains a SIPS URI, TLS MUST be used to communicate with that proxy.

   If the copy contains a Route header field, the proxy MUST inspect the URI in its first value. If that URI does not contain an lr parameter, the proxy MUST modify the copy as follows:





   - The proxy MUST place the Request-URI into the Route header field as the last value.

   - The proxy MUST then place the first Route header field value into the Request-URI and remove that value from the Route header field.

   Appending the Request-URI to the Route header field is part of a mechanism used to pass the information in that Request-URI through strict-routing elements. "Popping" the first Route header field value into the Request-URI formats the message the way a strict-routing element expects to receive it (with its own URI in the Request-URI and the next location to visit in the first Route header field value).

7. Determine Next-Hop Address, Port, and Transport

   The proxy MAY have a local policy to send the request to a specific IP address, port, and transport, independent of the values of the Route and Request-URI. Such a policy MUST NOT be

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 732
Keysight, EX-1045
Page 732

used if the proxy is not certain that the IP address, port, and transport correspond to a server that is a loose router. However, this mechanism for sending the request through a specific next hop is NOT RECOMMENDED; instead a Route header field should be used for that purpose as described above.

In the absence of such an overriding mechanism, the proxy applies the procedures listed in [4] as follows to determine where to send the request. If the proxy has reformatted the request to send to a strict-routing element as described in step 6 above, the proxy MUST apply those procedures to the Request-URI of the request. Otherwise, the proxy MUST apply the procedures to the first value in the Route header field, if present, else the Request-URI. The procedures will produce an ordered set of (address, port, transport) tuples. Independently of which URI is being used as input to the procedures of [4], if the Request-URI specifies a SIPS resource, the proxy MUST follow the procedures of [4] as if the input URI were a SIPS URI.

As described in [4], the proxy MUST attempt to deliver the message to the first tuple in that set, and proceed through the set in order until the delivery attempt succeeds.

For each tuple attempted, the proxy MUST format the message as appropriate for the tuple and send the request using a new client transaction as detailed in steps 8 through 10.





Since each attempt uses a new client transaction, it represents a new branch. Thus, the branch parameter provided with the Via header field inserted in step 8 MUST be different for each attempt.

If the client transaction reports failure to send the request or a timeout from its state machine, the proxy continues to the next address in that ordered set. If the ordered set is exhausted, the request cannot be forwarded to this element in the target set. The proxy does not need to place anything in the response context, but otherwise acts as if this element of the target set returned a 408 (Request Timeout) final response.

8. Add a Via header field value

The proxy MUST insert a Via header field value into the copy before the existing Via header field values. The construction of this value follows the same guidelines of Section 8.1.1.7. This implies that the proxy will compute its own branch parameter, which will be globally unique for that branch, and contain the requisite magic cookie. Note that this implies that the branch parameter will be different for different instances of a spiraled or looped request through a proxy.

Proxies choosing to detect loops have an additional constraint in the value they use for construction of the branch parameter. A proxy choosing to detect loops SHOULD create a branch

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 733
Keysight, EX-1045
Page 733

parameter separable into two parts by the implementation. The first part MUST satisfy the constraints of Section 8.1.1.7 as described above. The second is used to perform loop detection and distinguish loops from spirals.

Loop detection is performed by verifying that, when a request returns to a proxy, those fields having an impact on the processing of the request have not changed. The value placed in this part of the branch parameter SHOULD reflect all of those fields (including any Route, Proxy-Require and Proxy-Authorization header fields). This is to ensure that if the request is routed back to the proxy and one of those fields changes, it is treated as a spiral and not a loop (see Section 16.3). A common way to create this value is to compute a cryptographic hash of the To tag, From tag, Call-ID header field, the Request-URI of the request received (before translation), the topmost Via header, and the sequence number from the CSeq header field, in addition to any Proxy-Require and Proxy-Authorization header fields that may be present. The





algorithm used to compute the hash is implementation-dependent, but MD5 (RFC 1321 [35]), expressed in hexadecimal, is a reasonable choice. (Base64 is not permissible for a token.)

If a proxy wishes to detect loops, the "branch" parameter it supplies MUST depend on all information affecting processing of a request, including the incoming Request-URI and any header fields affecting the request's admission or routing. This is necessary to distinguish looped requests from requests whose routing parameters have changed before returning to this server.

The request method MUST NOT be included in the calculation of the branch parameter. In particular, CANCEL and ACK requests (for non-2xx responses) MUST have the same branch value as the corresponding request they cancel or acknowledge. The branch parameter is used in correlating those requests at the server handling them (see Sections 17.2.3 and 9.2).

9. Add a Content-Length header field if necessary

   If the request will be sent to the next hop using a stream-based transport and the copy contains no Content-Length header field, the proxy MUST insert one with the correct value for the body of the request (see Section 20.14).

10. Forward Request

    A stateful proxy MUST create a new client transaction for this request as described in Section 17.1 and instructs the transaction to send the request using the address, port and transport determined in step 7.

11. Set timer C

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 734
Keysight, EX-1045
Page 734

In order to handle the case where an INVITE request never generates a final response, the TU uses a timer which is called timer C. Timer C MUST be set for each client transaction when an INVITE request is proxied. The timer MUST be larger than 3 minutes. Section 16.7 bullet 2 discusses how this timer is updated with provisional responses, and Section 16.8 discusses processing when it fires.





16.7 Response Processing

When a response is received by an element, it first tries to locate a client transaction (Section 17.1.3) matching the response. If none is found, the element MUST process the response (even if it is an informational response) as a stateless proxy (described below). If a match is found, the response is handed to the client transaction.

Forwarding responses for which a client transaction (or more generally any knowledge of having sent an associated request) is not found improves robustness. In particular, it ensures that "late" 2xx responses to INVITE requests are forwarded properly.

As client transactions pass responses to the proxy layer, the following processing MUST take place:

1. Find the appropriate response context
2. Update timer C for provisional responses
3. Remove the topmost Via
4. Add the response to the response context
5. Check to see if this response should be forwarded immediately
6. When necessary, choose the best final response from the response context

If no final response has been forwarded after every client transaction associated with the response context has been terminated, the proxy must choose and forward the "best" response from those it has seen so far.

The following processing MUST be performed on each response that is forwarded. It is likely that more than one response to each request will be forwarded: at least each provisional and one final response.

7. Aggregate authorization header field values if necessary
8. Optionally rewrite Record-Route header field values

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

9. Forward the response

10. Generate any necessary CANCEL requests



Each of the above steps are detailed below:

1. Find Context

   The proxy locates the "response context" it created before forwarding the original request using the key described in Section 16.6. The remaining processing steps take place in this context.

2. Update timer C for provisional responses

   For an INVITE transaction, if the response is a provisional response with status codes 101 to 199 inclusive (i.e., anything but 100), the proxy MUST reset timer C for that client transaction. The timer MAY be reset to a different value, but this value MUST be greater than 3 minutes.

3. Via

   The proxy removes the topmost Via header field value from the response.

   If no Via header field values remain in the response, the response was meant for this element and MUST NOT be forwarded. The remainder of the processing described in this section is not performed on this message, the UAC processing rules described in Section 8.1.3 are followed instead (transport layer processing has already occurred).

   This will happen, for instance, when the element generates CANCEL requests as described in Section 10.

4. Add response to context

   Final responses received are stored in the response context until a final response is generated on the server transaction associated with this context. The response may be a candidate for the best final response to be returned on that server transaction. Information from this response may be needed in forming the best response, even if this response is not chosen.

   If the proxy chooses to recurse on any contacts in a 3xx response by adding them to the target set, it MUST remove them from the response before adding the response to the response context. However, a proxy SHOULD NOT recurse to a non-SIPS URI if the Request-URI of the original request was a SIPS URI. If

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

the proxy recurses on all of the contacts in a 3xx response, the proxy SHOULD NOT add the resulting contactless response to the response context.

Removing the contact before adding the response to the response context prevents the next element upstream from retrying a location this proxy has already attempted.

3xx responses may contain a mixture of SIP, SIPS, and non-SIP URIs. A proxy may choose to recurse on the SIP and SIPS URIs and place the remainder into the response context to be returned, potentially in the final response.

If a proxy receives a 416 (Unsupported URI Scheme) response to a request whose Request-URI scheme was not SIP, but the scheme in the original received request was SIP or SIPS (that is, the proxy changed the scheme from SIP or SIPS to something else when it proxied a request), the proxy SHOULD add a new URI to the target set. This URI SHOULD be a SIP URI version of the non-SIP URI that was just tried. In the case of the tel URL, this is accomplished by placing the telephone-subscriber part of the tel URL into the user part of the SIP URI, and setting the hostpart to the domain where the prior request was sent. See Section 19.1.6 for more detail on forming SIP URIs from tel URLs.

As with a 3xx response, if a proxy "recurses" on the 416 by trying a SIP or SIPS URI instead, the 416 response SHOULD NOT be added to the response context.

5. Check response for forwarding

Until a final response has been sent on the server transaction, the following responses MUST be forwarded immediately:

- Any provisional response other than 100 (Trying)

- Any 2xx response

If a 6xx response is received, it is not immediately forwarded, but the stateful proxy SHOULD cancel all client pending transactions as described in Section 10, and it MUST NOT create any new branches in this context.

This is a change from RFC 2543, which mandated that the proxy was to forward the 6xx response immediately. For an INVITE transaction, this approach had the problem that a 2xx response could arrive on another branch, in which case the proxy would



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 737
Keysight, EX-1045
Page 737

have to forward the 2xx. The result was that the UAC could receive a 6xx response followed by a 2xx response, which should never be allowed to happen. Under the new rules, upon receiving a 6xx, a proxy will issue a CANCEL request, which will generally result in 487 responses from all outstanding client transactions, and then at that point the 6xx is forwarded upstream.

After a final response has been sent on the server transaction, the following responses MUST be forwarded immediately:

- Any 2xx response to an INVITE request

A stateful proxy MUST NOT immediately forward any other responses. In particular, a stateful proxy MUST NOT forward any 100 (Trying) response. Those responses that are candidates for forwarding later as the "best" response have been gathered as described in step "Add Response to Context".

Any response chosen for immediate forwarding MUST be processed as described in steps "Aggregate Authorization Header Field Values" through "Record-Route".

This step, combined with the next, ensures that a stateful proxy will forward exactly one final response to a non-INVITE request, and either exactly one non-2xx response or one or more 2xx responses to an INVITE request.

6. Choosing the best response

A stateful proxy MUST send a final response to a response context's server transaction if no final responses have been immediately forwarded by the above rules and all client transactions in this response context have been terminated.

The stateful proxy MUST choose the "best" final response among those received and stored in the response context.

If there are no final responses in the context, the proxy MUST send a 408 (Request Timeout) response to the server transaction.

Otherwise, the proxy MUST forward a response from the responses stored in the response context. It MUST choose from the 6xx class responses if any exist in the context. If no 6xx class responses are present, the proxy SHOULD choose from the lowest response class stored in the response context. The proxy MAY select any response within that chosen class. The proxy SHOULD





give preference to responses that provide information affecting resubmission of this request, such as 401, 407, 415, 420, and 484 if the 4xx class is chosen.

A proxy which receives a 503 (Service Unavailable) response SHOULD NOT forward it upstream unless it can determine that any

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 738
Keysight, EX-1045
Page 738

subsequent requests it might proxy will also generate a 503. In other words, forwarding a 503 means that the proxy knows it cannot service any requests, not just the one for the Request-URI in the request which generated the 503. If the only response that was received is a 503, the proxy SHOULD generate a 500 response and forward that upstream.

The forwarded response MUST be processed as described in steps "Aggregate Authorization Header Field Values" through "Record-Route".

For example, if a proxy forwarded a request to 4 locations, and received 503, 407, 501, and 404 responses, it may choose to forward the 407 (Proxy Authentication Required) response.

1xx and 2xx responses may be involved in the establishment of dialogs. When a request does not contain a To tag, the To tag in the response is used by the UAC to distinguish multiple responses to a dialog creating request. A proxy MUST NOT insert a tag into the To header field of a 1xx or 2xx response if the request did not contain one. A proxy MUST NOT modify the tag in the To header field of a 1xx or 2xx response.

Since a proxy may not insert a tag into the To header field of a 1xx response to a request that did not contain one, it cannot issue non-100 provisional responses on its own. However, it can branch the request to a UAS sharing the same element as the proxy. This UAS can return its own provisional responses, entering into an early dialog with the initiator of the request. The UAS does not have to be a discreet process from the proxy. It could be a virtual UAS implemented in the same code space as the proxy.

3-6xx responses are delivered hop-by-hop. When issuing a 3-6xx response, the element is effectively acting as a UAS, issuing its own response, usually based on the responses received from downstream elements. An element SHOULD preserve the To tag when simply forwarding a 3-6xx response to a request that did not contain a To tag.

A proxy MUST NOT modify the To tag in any forwarded response to a request that contains a To tag.





While it makes no difference to the upstream elements if the proxy replaced the To tag in a forwarded 3-6xx response, preserving the original tag may assist with debugging.

When the proxy is aggregating information from several responses, choosing a To tag from among them is arbitrary, and generating a new To tag may make debugging easier. This happens, for instance, when combining 401 (Unauthorized) and 407 (Proxy Authentication Required) challenges, or combining Contact values from unencrypted and unauthenticated 3xx responses.

## 7. Aggregate Authorization Header Field Values

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 739
Keysight, EX-1045
Page 739

If the selected response is a 401 (Unauthorized) or 407 (Proxy Authentication Required), the proxy MUST collect any WWW-Authenticate and Proxy-Authenticate header field values from all other 401 (Unauthorized) and 407 (Proxy Authentication Required) responses received so far in this response context and add them to this response without modification before forwarding. The resulting 401 (Unauthorized) or 407 (Proxy Authentication Required) response could have several WWW-Authenticate AND Proxy-Authenticate header field values.

This is necessary because any or all of the destinations the request was forwarded to may have requested credentials. The client needs to receive all of those challenges and supply credentials for each of them when it retries the request. Motivation for this behavior is provided in Section 26.

8. Record-Route

If the selected response contains a Record-Route header field value originally provided by this proxy, the proxy MAY choose to rewrite the value before forwarding the response. This allows the proxy to provide different URIs for itself to the next upstream and downstream elements. A proxy may choose to use this mechanism for any reason. For instance, it is useful for multi-homed hosts.

If the proxy received the request over TLS, and sent it out over a non-TLS connection, the proxy MUST rewrite the URI in the Record-Route header field to be a SIPS URI. If the proxy received the request over a non-TLS connection, and sent it out over TLS, the proxy MUST rewrite the URI in the Record-Route header field to be a SIP URI.





The new URI provided by the proxy MUST satisfy the same constraints on URIs placed in Record-Route header fields in requests (see Step 4 of Section 16.6) with the following modifications:

The URI SHOULD NOT contain the transport parameter unless the proxy has knowledge that the next upstream (as opposed to downstream) element that will be in the path of subsequent requests supports that transport.

When a proxy does decide to modify the Record-Route header field in the response, one of the operations it performs is locating the Record-Route value that it had inserted. If the request spiraled, and the proxy inserted a Record-Route value in each iteration of the spiral, locating the correct value in the response (which must be the proper iteration in the reverse direction) is tricky. The rules above recommend that a proxy wishing to rewrite Record-Route header field values insert sufficiently distinct URIs into the Record-Route header field so that the right one may be selected for rewriting. A

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

RECOMMENDED mechanism to achieve this is for the proxy to append a unique identifier for the proxy instance to the user portion of the URI.

When the response arrives, the proxy modifies the first Record-Route whose identifier matches the proxy instance. The modification results in a URI without this piece of data appended to the user portion of the URI. Upon the next iteration, the same algorithm (find the topmost Record-Route header field value with the parameter) will correctly extract the next Record-Route header field value inserted by that proxy.

Not every response to a request to which a proxy adds a Record-Route header field value will contain a Record-Route header field. If the response does contain a Record-Route header field, it will contain the value the proxy added.

9. Forward response

After performing the processing described in steps "Aggregate Authorization Header Field Values" through "Record-Route", the proxy MAY perform any feature specific manipulations on the selected response. The proxy MUST NOT add to, modify, or remove the message body. Unless otherwise specified, the proxy MUST NOT remove any header field values other than the Via header field value discussed in Section 16.7 Item 3. In particular, the proxy MUST NOT remove any "received" parameter





it may have added to the next Via header field value while processing the request associated with this response. The proxy MUST pass the response to the server transaction associated with the response context. This will result in the response being sent to the location now indicated in the topmost Via header field value. If the server transaction is no longer available to handle the transmission, the element MUST forward the response statelessly by sending it to the server transport. The server transaction might indicate failure to send the response or signal a timeout in its state machine. These errors would be logged for diagnostic purposes as appropriate, but the protocol requires no remedial action from the proxy.

The proxy MUST maintain the response context until all of its associated transactions have been terminated, even after forwarding a final response.

10. Generate CANCELs

If the forwarded response was a final response, the proxy MUST generate a CANCEL request for all pending client transactions associated with this response context. A proxy SHOULD also generate a CANCEL request for all pending client transactions associated with this response context when it receives a 6xx response. A pending client transaction is one that has received a provisional response, but no final response (it is

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

in the proceeding state) and has not had an associated CANCEL generated for it. Generating CANCEL requests is described in Section 9.1.

The requirement to CANCEL pending client transactions upon forwarding a final response does not guarantee that an endpoint will not receive multiple 200 (OK) responses to an INVITE. 200 (OK) responses on more than one branch may be generated before the CANCEL requests can be sent and processed. Further, it is reasonable to expect that a future extension may override this requirement to issue CANCEL requests.

16.8 Processing Timer C

If timer C should fire, the proxy MUST either reset the timer with any value it chooses, or terminate the client transaction. If the client transaction has received a provisional response, the proxy MUST generate a CANCEL request matching that transaction. If the client transaction has not received a provisional response, the proxy MUST behave as if the transaction received a 408 (Request Timeout) response.





Allowing the proxy to reset the timer allows the proxy to dynamically extend the transaction's lifetime based on current conditions (such as utilization) when the timer fires.

16.9 Handling Transport Errors

If the transport layer notifies a proxy of an error when it tries to forward a request (see Section 18.4), the proxy MUST behave as if the forwarded request received a 503 (Service Unavailable) response.

If the proxy is notified of an error when forwarding a response, it drops the response. The proxy SHOULD NOT cancel any outstanding client transactions associated with this response context due to this notification.

If a proxy cancels its outstanding client transactions, a single malicious or misbehaving client can cause all transactions to fail through its Via header field.

16.10 CANCEL Processing

A stateful proxy MAY generate a CANCEL to any other request it has generated at any time (subject to receiving a provisional response to that request as described in section 9.1). A proxy MUST cancel any pending client transactions associated with a response context when it receives a matching CANCEL request.

A stateful proxy MAY generate CANCEL requests for pending INVITE client transactions based on the period specified in the INVITE's Expires header field elapsing. However, this is generally unnecessary since the endpoints involved will take care of signaling the end of the transaction.

While a CANCEL request is handled in a stateful proxy by its own

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

server transaction, a new response context is not created for it. Instead, the proxy layer searches its existing response contexts for the server transaction handling the request associated with this CANCEL. If a matching response context is found, the element MUST immediately return a 200 (OK) response to the CANCEL request. In this case, the element is acting as a user agent server as defined in Section 8.2. Furthermore, the element MUST generate CANCEL requests for all pending client transactions in the context as described in Section 16.7 step 10.

If a response context is not found, the element does not have any knowledge of the request to apply the CANCEL to. It MUST statelessly forward the CANCEL request (it may have statelessly forwarded the associated request previously).





### 16.11 Stateless Proxy

When acting statelessly, a proxy is a simple message forwarder. Much of the processing performed when acting statelessly is the same as when behaving statefully. The differences are detailed here.

A stateless proxy does not have any notion of a transaction, or of the response context used to describe stateful proxy behavior. Instead, the stateless proxy takes messages, both requests and responses, directly from the transport layer (See section 18). As a result, stateless proxies do not retransmit messages on their own. They do, however, forward all retransmissions they receive (they do not have the ability to distinguish a retransmission from the original message). Furthermore, when handling a request statelessly, an element MUST NOT generate its own 100 (Trying) or any other provisional response.

A stateless proxy MUST validate a request as described in Section 16.3

A stateless proxy MUST follow the request processing steps described in Sections 16.4 through 16.5 with the following exception:

- A stateless proxy MUST choose one and only one target from the target set. This choice MUST only rely on fields in the message and time-invariant properties of the server. In particular, a retransmitted request MUST be forwarded to the same destination each time it is processed. Furthermore, CANCEL and non-Routed ACK requests MUST generate the same choice as their associated INVITE.

A stateless proxy MUST follow the request processing steps described in Section 16.6 with the following exceptions:

- The requirement for unique branch IDs across space and time applies to stateless proxies as well. However, a stateless proxy cannot simply use a random number generator to compute the first component of the branch ID, as described in Section 16.6 bullet 8. This is because retransmissions of a request need to have the same value, and a stateless proxy cannot tell a retransmission from the original request. Therefore, the

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 743
Keysight, EX-1045
Page 743

component of the branch parameter that makes it unique MUST be the same each time a retransmitted request is forwarded. Thus for a stateless proxy, the branch parameter MUST be computed as a combinatoric function of message parameters which are invariant on retransmission.



The stateless proxy MAY use any technique it likes to guarantee uniqueness of its branch IDs across transactions. However, the following procedure is RECOMMENDED. The proxy examines the branch ID in the topmost Via header field of the received request. If it begins with the magic cookie, the first component of the branch ID of the outgoing request is computed as a hash of the received branch ID. Otherwise, the first component of the branch ID is computed as a hash of the topmost Via, the tag in the To header field, the tag in the From header field, the Call-ID header field, the CSeq number (but not method), and the Request-URI from the received request. One of these fields will always vary across two different transactions.

- All other message transformations specified in Section 16.6 MUST result in the same transformation of a retransmitted request. In particular, if the proxy inserts a Record-Route value or pushes URIs into the Route header field, it MUST place the same values in retransmissions of the request. As for the Via branch parameter, this implies that the transformations MUST be based on time-invariant configuration or retransmission-invariant properties of the request.

- A stateless proxy determines where to forward the request as described for stateful proxies in Section 16.6 Item 10. The request is sent directly to the transport layer instead of through a client transaction.

  Since a stateless proxy must forward retransmitted requests to the same destination and add identical branch parameters to each of them, it can only use information from the message itself and time-invariant configuration data for those calculations. If the configuration state is not time-invariant (for example, if a routing table is updated) any requests that could be affected by the change may not be forwarded statelessly during an interval equal to the transaction timeout window before or after the change. The method of processing the affected requests in that interval is an implementation decision. A common solution is to forward them transaction statefully.

Stateless proxies MUST NOT perform special processing for CANCEL requests. They are processed by the above rules as any other requests. In particular, a stateless proxy applies the same Route header field processing to CANCEL requests that it applies to any other request.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 744
Keysight, EX-1045
Page 744

Response processing as described in Section 16.7 does not apply to a proxy behaving statelessly. When a response arrives at a stateless proxy, the proxy MUST inspect the sent-by value in the first (topmost) Via header field value. If that address matches the proxy, (it equals a value this proxy has inserted into previous requests) the proxy MUST remove that header field value from the response and forward the result to the location indicated in the next Via header field value. The proxy MUST NOT add to, modify, or remove the message body. Unless specified otherwise, the proxy MUST NOT remove any other header field values. If the address does not match the proxy, the message MUST be silently discarded.

### 16.12 Summary of Proxy Route Processing

In the absence of local policy to the contrary, the processing a proxy performs on a request containing a Route header field can be summarized in the following steps.

1. The proxy will inspect the Request-URI. If it indicates a resource owned by this proxy, the proxy will replace it with the results of running a location service. Otherwise, the proxy will not change the Request-URI.

2. The proxy will inspect the URI in the topmost Route header field value. If it indicates this proxy, the proxy removes it from the Route header field (this route node has been reached).

3. The proxy will forward the request to the resource indicated by the URI in the topmost Route header field value or in the Request-URI if no Route header field is present. The proxy determines the address, port and transport to use when forwarding the request by applying the procedures in [4] to that URI.

If no strict-routing elements are encountered on the path of the request, the Request-URI will always indicate the target of the request.

#### 16.12.1 Examples

##### 16.12.1.1 Basic SIP Trapezoid

This scenario is the basic SIP trapezoid, U1 -> P1 -> P2 -> U2, with both proxies record-routing. Here is the flow.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 745
Keysight, EX-1045
Page 745

U1 sends:

```
INVITE sip:callee@domain.com SIP/2.0
Contact: sip:caller@u1.example.com
```

to P1. P1 is an outbound proxy. P1 is not responsible for domain.com, so it looks it up in DNS and sends it there. It also adds a Record-Route header field value:

```
INVITE sip:callee@domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p1.example.com;lr>
```

P2 gets this. It is responsible for domain.com so it runs a location service and rewrites the Request-URI. It also adds a Record-Route header field value. There is no Route header field, so it resolves the new Request-URI to determine where to send the request:

```
INVITE sip:callee@u2.domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p2.domain.com;lr>
Record-Route: <sip:p1.example.com;lr>
```

The callee at u2.domain.com gets this and responds with a 200 OK:

```
SIP/2.0 200 OK
Contact: sip:callee@u2.domain.com
Record-Route: <sip:p2.domain.com;lr>
Record-Route: <sip:p1.example.com;lr>
```

The callee at u2 also sets its dialog state's remote target URI to sip:caller@u1.example.com and its route set to:

```
(<sip:p2.domain.com;lr>,<sip:p1.example.com;lr>)
```

This is forwarded by P2 to P1 to U1 as normal. Now, U1 sets its dialog state's remote target URI to sip:callee@u2.domain.com and its route set to:

```
(<sip:p1.example.com;lr>,<sip:p2.domain.com;lr>)
```

Since all the route set elements contain the lr parameter, U1 constructs the following BYE request:

```
BYE sip:callee@u2.domain.com SIP/2.0
Route: <sip:p1.example.com;lr>,<sip:p2.domain.com;lr>
```





As any other element (including proxies) would do, it resolves the URI in the topmost Route header field value using DNS to determine where to send the request. This goes to P1. P1 notices that it is not responsible for the resource indicated in the Request-URI so it doesn't change it. It does see that it is the first value in the Route header field, so it removes that value, and forwards the

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

request to P2:

```
BYE sip:callee@u2.domain.com SIP/2.0
Route: <sip:p2.domain.com;lr>
```

P2 also notices it is not responsible for the resource indicated by the Request-URI (it is responsible for domain.com, not u2.domain.com), so it doesn't change it. It does see itself in the first Route header field value, so it removes it and forwards the following to u2.domain.com based on a DNS lookup against the Request-URI:

```
BYE sip:callee@u2.domain.com SIP/2.0
```

#### 16.12.1.2 Traversing a Strict-Routing Proxy

In this scenario, a dialog is established across four proxies, each of which adds Record-Route header field values. The third proxy implements the strict-routing procedures specified in RFC 2543 and many works in progress.

```
U1->P1->P2->P3->P4->U2
```

The INVITE arriving at U2 contains:

```
INVITE sip:callee@u2.domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p4.domain.com;lr>
Record-Route: <sip:p3.middle.com>
Record-Route: <sip:p2.example.com;lr>
Record-Route: <sip:p1.example.com;lr>
```

Which U2 responds to with a 200 OK. Later, U2 sends the following BYE request to P4 based on the first Route header field value.

```
BYE sip:caller@u1.example.com SIP/2.0
Route: <sip:p4.domain.com;lr>
Route: <sip:p3.middle.com>
Route: <sip:p2.example.com;lr>
Route: <sip:p1.example.com;lr>
```





P4 is not responsible for the resource indicated in the Request-URI so it will leave it alone. It notices that it is the element in the first Route header field value so it removes it. It then prepares to send the request based on the now first Route header field value of sip:p3.middle.com, but it notices that this URI does not contain the lr parameter, so before sending, it reformats the request to be:

```
BYE sip:p3.middle.com SIP/2.0
Route: <sip:p2.example.com;lr>
Route: <sip:p1.example.com;lr>
Route: <sip:caller@u1.example.com>
```

P3 is a strict router, so it forwards the following to P2:

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 747
Keysight, EX-1045
Page 747

```
   BYE sip:p2.example.com;lr SIP/2.0
   Route: <sip:p1.example.com;lr>
   Route: <sip:caller@u1.example.com>
```

P2 sees the request-URI is a value it placed into a Record-Route header field, so before further processing, it rewrites the request to be:

```
   BYE sip:caller@u1.example.com SIP/2.0
   Route: <sip:p1.example.com;lr>
```

P2 is not responsible for u1.example.com, so it sends the request to P1 based on the resolution of the Route header field value.

P1 notices itself in the topmost Route header field value, so it removes it, resulting in:

```
   BYE sip:caller@u1.example.com SIP/2.0
```

Since P1 is not responsible for u1.example.com and there is no Route header field, P1 will forward the request to u1.example.com based on the Request-URI.

#### 16.12.1.3 Rewriting Record-Route Header Field Values

In this scenario, U1 and U2 are in different private namespaces and they enter a dialog through a proxy P1, which acts as a gateway between the namespaces.

```
   U1->P1->U2
```





U1 sends:

```
   INVITE sip:callee@gateway.leftprivatespace.com SIP/2.0
   Contact: <sip:caller@u1.leftprivatespace.com>
```

P1 uses its location service and sends the following to U2:

```
   INVITE sip:callee@rightprivatespace.com SIP/2.0
   Contact: <sip:caller@u1.leftprivatespace.com>
   Record-Route: <sip:gateway.rightprivatespace.com;lr>
```

U2 sends this 200 (OK) back to P1:

```
   SIP/2.0 200 OK
   Contact: <sip:callee@u2.rightprivatespace.com>
   Record-Route: <sip:gateway.rightprivatespace.com;lr>
```

P1 rewrites its Record-Route header parameter to provide a value that U1 will find useful, and sends the following to U1:

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 748
Keysight, EX-1045
Page 748

```
      SIP/2.0 200 OK
      Contact: <sip:callee@u2.rightprivatespace.com>
      Record-Route: <sip:gateway.leftprivatespace.com;lr>
```

Later, U1 sends the following BYE request to P1:

```
      BYE sip:callee@u2.rightprivatespace.com SIP/2.0
      Route: <sip:gateway.leftprivatespace.com;lr>
```

which P1 forwards to U2 as:

```
      BYE sip:callee@u2.rightprivatespace.com SIP/2.0
```

# 17 Transactions

SIP is a transactional protocol: interactions between components take place in a series of independent message exchanges. Specifically, a SIP transaction consists of a single request and any responses to that request, which include zero or more provisional responses and one or more final responses. In the case of a transaction where the request was an INVITE (known as an INVITE transaction), the transaction also includes the ACK only if the final response was not a 2xx response. If the response was a 2xx, the ACK is not considered part of the transaction.

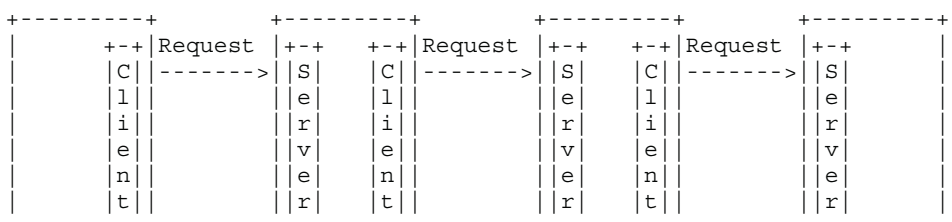
> The reason for this separation is rooted in the importance of delivering all 200 (OK) responses to an INVITE to the UAC. To deliver them all to the UAC, the UAS alone takes responsibility





> for retransmitting them (see Section 13.3.1.4), and the UAC alone takes responsibility for acknowledging them with ACK (see Section 13.2.2.4). Since this ACK is retransmitted only by the UAC, it is effectively considered its own transaction.

Transactions have a client side and a server side. The client side is known as a client transaction and the server side as a server transaction. The client transaction sends the request, and the server transaction sends the response. The client and server transactions are logical functions that are embedded in any number of elements. Specifically, they exist within user agents and stateful proxy servers. Consider the example in Section 4. In this example, the UAC executes the client transaction, and its outbound proxy executes the server transaction. The outbound proxy also executes a client transaction, which sends the request to a server transaction in the inbound proxy. That proxy also executes a client transaction, which in turn sends the request to a server transaction in the UAS. This is shown in Figure 4.




https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 749
Keysight, EX-1045
Page 749

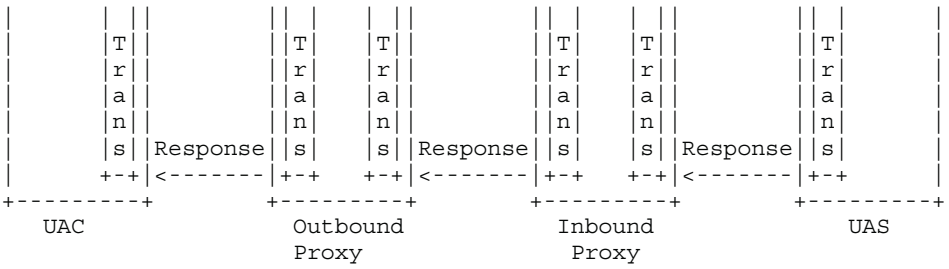



Figure 4: Transaction relationships

A stateless proxy does not contain a client or server transaction. The transaction exists between the UA or stateful proxy on one side, and the UA or stateful proxy on the other side. As far as SIP transactions are concerned, stateless proxies are effectively transparent. The purpose of the client transaction is to receive a request from the element in which the client is embedded (call this element the "Transaction User" or TU; it can be a UA or a stateful proxy), and reliably deliver the request to a server transaction.





The client transaction is also responsible for receiving responses and delivering them to the TU, filtering out any response retransmissions or disallowed responses (such as a response to ACK). Additionally, in the case of an INVITE request, the client transaction is responsible for generating the ACK request for any final response accepting a 2xx response.

Similarly, the purpose of the server transaction is to receive requests from the transport layer and deliver them to the TU. The server transaction filters any request retransmissions from the network. The server transaction accepts responses from the TU and delivers them to the transport layer for transmission over the network. In the case of an INVITE transaction, it absorbs the ACK request for any final response excepting a 2xx response.

The 2xx response and its ACK receive special treatment. This response is retransmitted only by a UAS, and its ACK generated only by the UAC. This end-to-end treatment is needed so that a caller knows the entire set of users that have accepted the call. Because of this special handling, retransmissions of the 2xx response are handled by the UA core, not the transaction layer. Similarly, generation of the ACK for the 2xx is handled by the UA core. Each proxy along the path merely forwards each 2xx response to INVITE and its corresponding ACK.

## 17.1 Client Transaction

The client transaction provides its functionality through the maintenance of a state machine.

The TU communicates with the client transaction through a simple interface. When the TU wishes to initiate a new transaction, it creates a client transaction and passes it the SIP request to send and an IP address, port, and transport to which to send it. The

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 750
Keysight, EX-1045
Page 750

client transaction begins execution of its state machine. Valid responses are passed up to the TU from the client transaction.

There are two types of client transaction state machines, depending on the method of the request passed by the TU. One handles client transactions for INVITE requests. This type of machine is referred to as an INVITE client transaction. Another type handles client transactions for all requests except INVITE and ACK. This is referred to as a non-INVITE client transaction. There is no client transaction for ACK. If the TU wishes to send an ACK, it passes one directly to the transport layer for transmission.





The INVITE transaction is different from those of other methods because of its extended duration. Normally, human input is required in order to respond to an INVITE. The long delays expected for sending a response argue for a three-way handshake. On the other hand, requests of other methods are expected to complete rapidly. Because of the non-INVITE transaction's reliance on a two-way handshake, TUs SHOULD respond immediately to non-INVITE requests.

### 17.1.1 INVITE Client Transaction

#### 17.1.1.1 Overview of INVITE Transaction

The INVITE transaction consists of a three-way handshake. The client transaction sends an INVITE, the server transaction sends responses, and the client transaction sends an ACK. For unreliable transports (such as UDP), the client transaction retransmits requests at an interval that starts at T1 seconds and doubles after every retransmission. T1 is an estimate of the round-trip time (RTT), and it defaults to 500 ms. Nearly all of the transaction timers described here scale with T1, and changing T1 adjusts their values. The request is not retransmitted over reliable transports. After receiving a 1xx response, any retransmissions cease altogether, and the client waits for further responses. The server transaction can send additional 1xx responses, which are not transmitted reliably by the server transaction. Eventually, the server transaction decides to send a final response. For unreliable transports, that response is retransmitted periodically, and for reliable transports, it is sent once. For each final response that is received at the client transaction, the client transaction sends an ACK, the purpose of which is to quench retransmissions of the response.

#### 17.1.1.2 Formal Description

The state machine for the INVITE client transaction is shown in Figure 5. The initial state, "calling", MUST be entered when the TU initiates a new client transaction with an INVITE request. The client transaction MUST pass the request to the transport layer for transmission (see Section 18). If an unreliable transport is being used, the client transaction MUST start timer A with a value of T1. If a reliable transport is being used, the client transaction SHOULD NOT start timer A (Timer A controls request retransmissions). For

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 751
Keysight, EX-1045
Page 751

any transport, the client transaction MUST start timer B with a value of 64*T1 seconds (Timer B controls transaction timeouts).

When timer A fires, the client transaction MUST retransmit the request by passing it to the transport layer, and MUST reset the timer with a value of 2*T1. The formal definition of retransmit



within the context of the transaction layer is to take the message previously sent to the transport layer and pass it to the transport layer once more.

When timer A fires 2*T1 seconds later, the request MUST be retransmitted again (assuming the client transaction is still in this state). This process MUST continue so that the request is retransmitted with intervals that double after each transmission. These retransmissions SHOULD only be done while the client transaction is in the "calling" state.

The default value for T1 is 500 ms. T1 is an estimate of the RTT between the client and server transactions. Elements MAY (though it is NOT RECOMMENDED) use smaller values of T1 within closed, private networks that do not permit general Internet connection. T1 MAY be chosen larger, and this is RECOMMENDED if it is known in advance (such as on high latency access links) that the RTT is larger. Whatever the value of T1, the exponential backoffs on retransmissions described in this section MUST be used.

If the client transaction is still in the "Calling" state when timer B fires, the client transaction SHOULD inform the TU that a timeout has occurred. The client transaction MUST NOT generate an ACK. The value of 64*T1 is equal to the amount of time required to send seven requests in the case of an unreliable transport.

If the client transaction receives a provisional response while in the "Calling" state, it transitions to the "Proceeding" state. In the "Proceeding" state, the client transaction SHOULD NOT retransmit the request any longer. Furthermore, the provisional response MUST be passed to the TU. Any further provisional responses MUST be passed up to the TU while in the "Proceeding" state.

When in either the "Calling" or "Proceeding" states, reception of a response with status code from 300-699 MUST cause the client transaction to transition to "Completed". The client transaction MUST pass the received response up to the TU, and the client transaction MUST generate an ACK request, even if the transport is reliable (guidelines for constructing the ACK from the response are given in Section 17.1.1.3) and then pass the ACK to the transport layer for transmission. The ACK MUST be sent to the same address, port, and transport to which the original request was sent. The client transaction SHOULD start timer D when it enters the "Completed" state, with a value of at least 32 seconds for unreliable transports, and a value of zero seconds for reliable transports. Timer D reflects the amount of time that the server transaction can remain in the "Completed" state when unreliable transports are used. This is equal to Timer H in the INVITE server transaction, whose

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

default is 64*T1. However, the client transaction does not know the value of T1 in use by the server transaction, so an absolute minimum of 32s is used instead of basing Timer D on T1.

Any retransmissions of the final response that are received while in the "Completed" state MUST cause the ACK to be re-passed to the transport layer for retransmission, but the newly received response MUST NOT be passed up to the TU. A retransmission of the response is defined as any response which would match the same client transaction based on the rules of Section 17.1.3.





https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 753
Keysight, EX-1045
Page 753

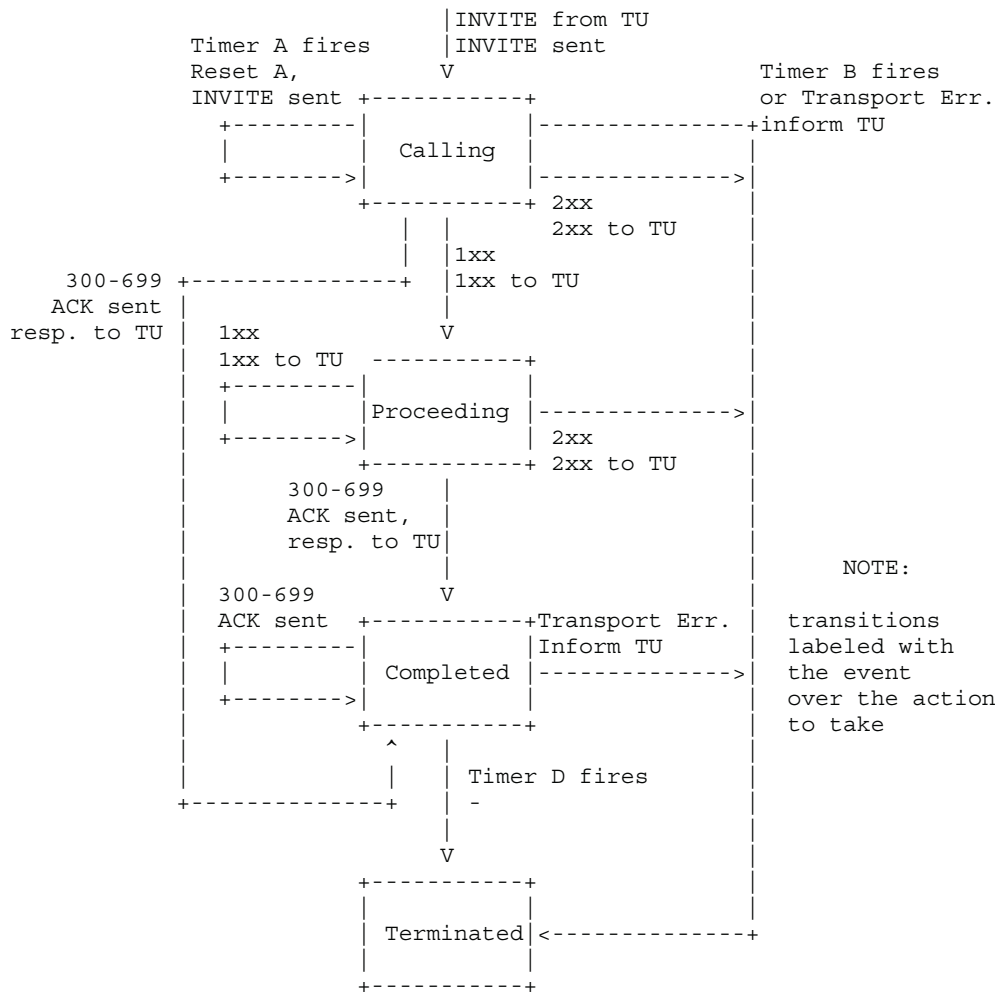



Figure 5: INVITE client transaction

If timer D fires while the client transaction is in the "Completed" state, the client transaction MUST move to the terminated state.

When in either the "Calling" or "Proceeding" states, reception of a 2xx response MUST cause the client transaction to enter the "Terminated" state, and the response MUST be passed up to the TU. The handling of this response depends on whether the TU is a proxy





core or a UAC core. A UAC core will handle generation of the ACK for this response, while a proxy core will always forward the 200 (OK) upstream. The differing treatment of 200 (OK) between proxy and UAC is the reason that handling of it does not take place in the transaction layer.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 754
Keysight, EX-1045
Page 754

The client transaction MUST be destroyed the instant it enters the "Terminated" state. This is actually necessary to guarantee correct operation. The reason is that 2xx responses to an INVITE are treated differently; each one is forwarded by proxies, and the ACK handling in a UAC is different. Thus, each 2xx needs to be passed to a proxy core (so that it can be forwarded) and to a UAC core (so it can be acknowledged). No transaction layer processing takes place. Whenever a response is received by the transport, if the transport layer finds no matching client transaction (using the rules of Section 17.1.3), the response is passed directly to the core. Since the matching client transaction is destroyed by the first 2xx, subsequent 2xx will find no match and therefore be passed to the core.

#### 17.1.1.3 Construction of the ACK Request

This section specifies the construction of ACK requests sent within the client transaction. A UAC core that generates an ACK for 2xx MUST instead follow the rules described in Section 13.

The ACK request constructed by the client transaction MUST contain values for the Call-ID, From, and Request-URI that are equal to the values of those header fields in the request passed to the transport by the client transaction (call this the "original request"). The To header field in the ACK MUST equal the To header field in the response being acknowledged, and therefore will usually differ from the To header field in the original request by the addition of the tag parameter. The ACK MUST contain a single Via header field, and this MUST be equal to the top Via header field of the original request. The CSeq header field in the ACK MUST contain the same value for the sequence number as was present in the original request, but the method parameter MUST be equal to "ACK".





If the INVITE request whose response is being acknowledged had Route header fields, those header fields MUST appear in the ACK. This is to ensure that the ACK can be routed properly through any downstream stateless proxies.

Although any request MAY contain a body, a body in an ACK is special since the request cannot be rejected if the body is not understood. Therefore, placement of bodies in ACK for non-2xx is NOT RECOMMENDED, but if done, the body types are restricted to any that appeared in the INVITE, assuming that the response to the INVITE was not 415. If it was, the body in the ACK MAY be any type listed in the Accept header field in the 415.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

For example, consider the following request:

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKkjshdyff
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=88sja8x
Max-Forwards: 70
Call-ID: 987asjd97y7atg
CSeq: 986759 INVITE
```

The ACK request for a non-2xx final response to this request would look like this:

```
ACK sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKkjshdyff
To: Bob <sip:bob@biloxi.com>;tag=99sa0xk
From: Alice <sip:alice@atlanta.com>;tag=88sja8x
Max-Forwards: 70
Call-ID: 987asjd97y7atg
CSeq: 986759 ACK
```

### 17.1.2 Non-INVITE Client Transaction

#### 17.1.2.1 Overview of the non-INVITE Transaction

Non-INVITE transactions do not make use of ACK. They are simple request-response interactions. For unreliable transports, requests are retransmitted at an interval which starts at T1 and doubles until it hits T2. If a provisional response is received, retransmissions continue for unreliable transports, but at an interval of T2. The server transaction retransmits the last response it sent, which can be a provisional or final response, only when a retransmission of the request is received. This is why request retransmissions need to continue even after a provisional response; they are to ensure reliable delivery of the final response.





Unlike an INVITE transaction, a non-INVITE transaction has no special handling for the 2xx response. The result is that only a single 2xx response to a non-INVITE is ever delivered to a UAC.

#### 17.1.2.2 Formal Description

The state machine for the non-INVITE client transaction is shown in Figure 6. It is very similar to the state machine for INVITE.

The "Trying" state is entered when the TU initiates a new client transaction with a request. When entering this state, the client transaction SHOULD set timer F to fire in 64*T1 seconds. The request MUST be passed to the transport layer for transmission. If an unreliable transport is in use, the client transaction MUST set timer E to fire in T1 seconds. If timer E fires while still in this state, the timer is reset, but this time with a value of MIN(2*T1, T2). When the timer fires again, it is reset to a MIN(4*T1, T2). This process continues so that retransmissions occur with an exponentially increasing interval that caps at T2. The default value of T2 is 4s, and it represents the amount of time a non-INVITE server transaction

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

will take to respond to a request, if it does not respond immediately. For the default values of T1 and T2, this results in intervals of 500 ms, 1 s, 2 s, 4 s, 4 s, 4 s, etc.

If Timer F fires while the client transaction is still in the "Trying" state, the client transaction SHOULD inform the TU about the timeout, and then it SHOULD enter the "Terminated" state. If a provisional response is received while in the "Trying" state, the response MUST be passed to the TU, and then the client transaction SHOULD move to the "Proceeding" state. If a final response (status codes 200-699) is received while in the "Trying" state, the response MUST be passed to the TU, and the client transaction MUST transition to the "Completed" state.

If Timer E fires while in the "Proceeding" state, the request MUST be passed to the transport layer for retransmission, and Timer E MUST be reset with a value of T2 seconds. If timer F fires while in the "Proceeding" state, the TU MUST be informed of a timeout, and the client transaction MUST transition to the terminated state. If a final response (status codes 200-699) is received while in the "Proceeding" state, the response MUST be passed to the TU, and the client transaction MUST transition to the "Completed" state.

Once the client transaction enters the "Completed" state, it MUST set Timer K to fire in T4 seconds for unreliable transports, and zero seconds for reliable transports. The "Completed" state exists to buffer any additional response retransmissions that may be received (which is why the client transaction remains there only for





unreliable transports). T4 represents the amount of time the network will take to clear messages between client and server transactions. The default value of T4 is 5s. A response is a retransmission when it matches the same transaction, using the rules specified in Section 17.1.3. If Timer K fires while in this state, the client transaction MUST transition to the "Terminated" state.

Once the transaction is in the terminated state, it MUST be destroyed immediately.

### 17.1.3 Matching Responses to Client Transactions

When the transport layer in the client receives a response, it has to determine which client transaction will handle the response, so that the processing of Sections 17.1.1 and 17.1.2 can take place. The branch parameter in the top Via header field is used for this purpose. A response matches a client transaction under two conditions:

1. If the response has the same value of the branch parameter in the top Via header field as the branch parameter in the top Via header field of the request that created the transaction.

2. If the method parameter in the CSeq header field matches the method of the request that created the transaction. The method is needed since a CANCEL request constitutes a different transaction, but shares the same value of the branch

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 757
Keysight, EX-1045
Page 757

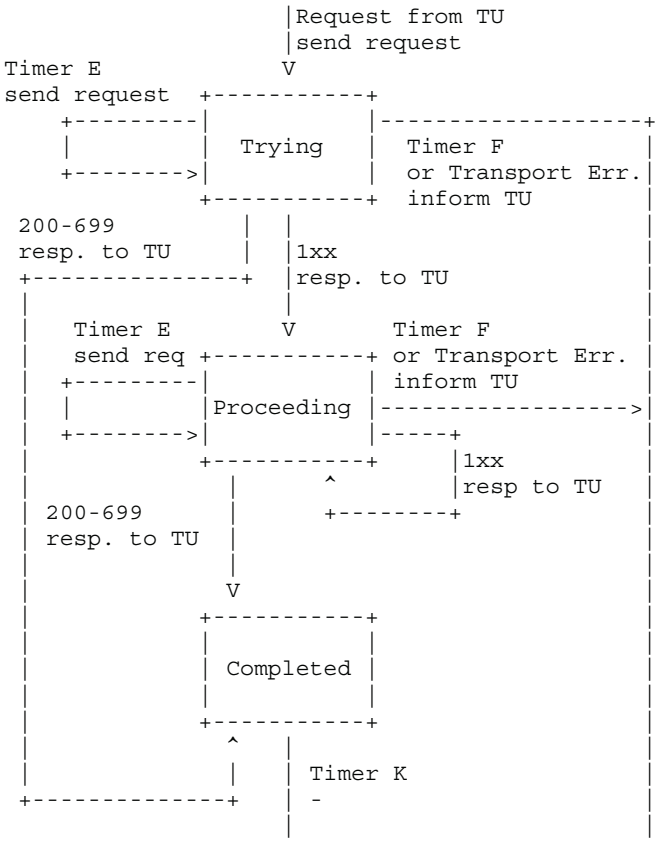
parameter.

If a request is sent via multicast, it is possible that it will generate multiple responses from different servers. These responses will all have the same branch parameter in the topmost Via, but vary in the To tag. The first response received, based on the rules above, will be used, and others will be viewed as retransmissions. That is not an error; multicast SIP provides only a rudimentary "single-hop-discovery-like" service that is limited to processing a single response. See Section 18.1.1 for details.





17.1.4 Handling Transport Errors




https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 758
Keysight, EX-1045
Page 758

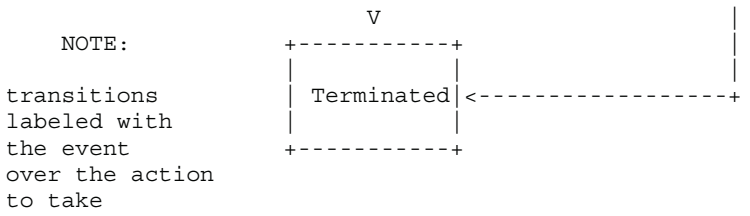



Figure 6: non-INVITE client transaction

When the client transaction sends a request to the transport layer to be sent, the following procedures are followed if the transport layer indicates a failure.





The client transaction SHOULD inform the TU that a transport failure has occurred, and the client transaction SHOULD transition directly to the "Terminated" state. The TU will handle the failover mechanisms described in [4].

### 17.2 Server Transaction

The server transaction is responsible for the delivery of requests to the TU and the reliable transmission of responses. It accomplishes this through a state machine. Server transactions are created by the core when a request is received, and transaction handling is desired for that request (this is not always the case).

As with the client transactions, the state machine depends on whether the received request is an INVITE request.

#### 17.2.1 INVITE Server Transaction

The state diagram for the INVITE server transaction is shown in Figure 7.

When a server transaction is constructed for a request, it enters the "Proceeding" state. The server transaction MUST generate a 100 (Trying) response unless it knows that the TU will generate a provisional or final response within 200 ms, in which case it MAY generate a 100 (Trying) response. This provisional response is needed to quench request retransmissions rapidly in order to avoid network congestion. The 100 (Trying) response is constructed according to the procedures in Section 8.2.6, except that the insertion of tags in the To header field of the response (when none was present in the request) is downgraded from MAY to SHOULD NOT. The request MUST be passed to the TU.

The TU passes any number of provisional responses to the server transaction. So long as the server transaction is in the "Proceeding" state, each of these MUST be passed to the transport layer for transmission. They are not sent reliably by the transaction layer (they are not retransmitted by it) and do not cause a change in the state of the server transaction. If a request retransmission is received while in the "Proceeding" state, the most recent provisional response that was received from the TU MUST be

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 759
Keysight, EX-1045
Page 759

passed to the transport layer for retransmission. A request is a retransmission if it matches the same server transaction based on the rules of Section 17.2.3.

If, while in the "Proceeding" state, the TU passes a 2xx response to the server transaction, the server transaction MUST pass this response to the transport layer for transmission. It is not





retransmitted by the server transaction; retransmissions of 2xx responses are handled by the TU. The server transaction MUST then transition to the "Terminated" state.

While in the "Proceeding" state, if the TU passes a response with status code from 300 to 699 to the server transaction, the response MUST be passed to the transport layer for transmission, and the state machine MUST enter the "Completed" state. For unreliable transports, timer G is set to fire in T1 seconds, and is not set to fire for reliable transports.

> This is a change from RFC 2543, where responses were always retransmitted, even over reliable transports.

When the "Completed" state is entered, timer H MUST be set to fire in 64*T1 seconds for all transports. Timer H determines when the server transaction abandons retransmitting the response. Its value is chosen to equal Timer B, the amount of time a client transaction will continue to retry sending a request. If timer G fires, the response is passed to the transport layer once more for retransmission, and timer G is set to fire in MIN(2*T1, T2) seconds. From then on, when timer G fires, the response is passed to the transport again for transmission, and timer G is reset with a value that doubles, unless that value exceeds T2, in which case it is reset with the value of T2. This is identical to the retransmit behavior for requests in the "Trying" state of the non-INVITE client transaction. Furthermore, while in the "Completed" state, if a request retransmission is received, the server SHOULD pass the response to the transport for retransmission.

If an ACK is received while the server transaction is in the "Completed" state, the server transaction MUST transition to the "Confirmed" state. As Timer G is ignored in this state, any retransmissions of the response will cease.

If timer H fires while in the "Completed" state, it implies that the ACK was never received. In this case, the server transaction MUST transition to the "Terminated" state, and MUST indicate to the TU that a transaction failure has occurred.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

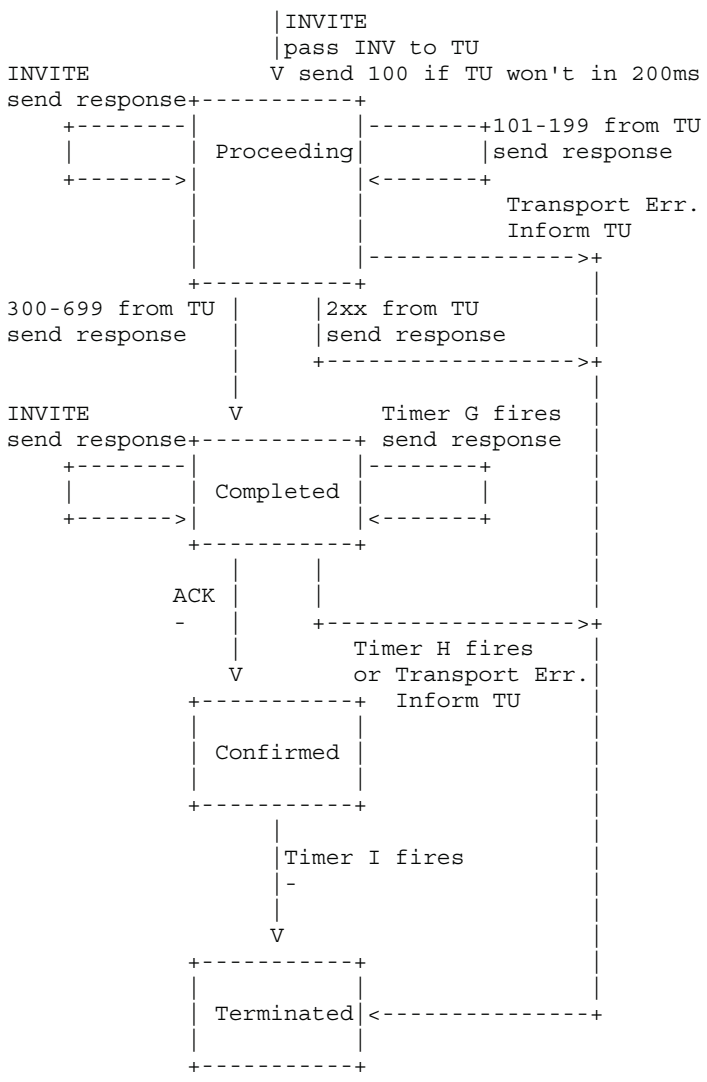



Figure 7: INVITE server transaction





https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 761
Keysight, EX-1045
Page 761

The purpose of the "Confirmed" state is to absorb any additional ACK messages that arrive, triggered from retransmissions of the final response. When this state is entered, timer I is set to fire in T4 seconds for unreliable transports, and zero seconds for reliable transports. Once timer I fires, the server MUST transition to the "Terminated" state.

Once the transaction is in the "Terminated" state, it MUST be destroyed immediately. As with client transactions, this is needed to ensure reliability of the 2xx responses to INVITE.

### 17.2.2 Non-INVITE Server Transaction

The state machine for the non-INVITE server transaction is shown in Figure 8.

The state machine is initialized in the "Trying" state and is passed a request other than INVITE or ACK when initialized. This request is passed up to the TU. Once in the "Trying" state, any further request retransmissions are discarded. A request is a retransmission if it matches the same server transaction, using the rules specified in Section 17.2.3.

While in the "Trying" state, if the TU passes a provisional response to the server transaction, the server transaction MUST enter the "Proceeding" state. The response MUST be passed to the transport layer for transmission. Any further provisional responses that are received from the TU while in the "Proceeding" state MUST be passed to the transport layer for transmission. If a retransmission of the request is received while in the "Proceeding" state, the most recently sent provisional response MUST be passed to the transport layer for retransmission. If the TU passes a final response (status codes 200-699) to the server while in the "Proceeding" state, the transaction MUST enter the "Completed" state, and the response MUST be passed to the transport layer for transmission.

When the server transaction enters the "Completed" state, it MUST set Timer J to fire in 64*T1 seconds for unreliable transports, and zero seconds for reliable transports. While in the "Completed" state, the server transaction MUST pass the final response to the transport layer for retransmission whenever a retransmission of the request is received. Any other final responses passed by the TU to the server transaction MUST be discarded while in the "Completed" state. The server transaction remains in this state until Timer J fires, at which point it MUST transition to the "Terminated" state.

The server transaction MUST be destroyed the instant it enters the "Terminated" state.



### 17.2.3 Matching Requests to Server Transactions

When a request is received from the network by the server, it has to be matched to an existing transaction. This is accomplished in the following manner.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 762
Keysight, EX-1045
Page 762

The branch parameter in the topmost Via header field of the request is examined. If it is present and begins with the magic cookie "z9hG4bK", the request was generated by a client transaction compliant to this specification. Therefore, the branch parameter will be unique across all transactions sent by that client. The request matches a transaction if:

1. the branch parameter in the request is equal to the one in the top Via header field of the request that created the transaction, and

2. the sent-by value in the top Via of the request is equal to the one in the request that created the transaction, and

3. the method of the request matches the one that created the transaction, except for ACK, where the method of the request that created the transaction is INVITE.

This matching rule applies to both INVITE and non-INVITE transactions alike.

The sent-by value is used as part of the matching process because there could be accidental or malicious duplication of branch parameters from different clients.

If the branch parameter in the top Via header field is not present, or does not contain the magic cookie, the following procedures are used. These exist to handle backwards compatibility with RFC 2543 compliant implementations.

The INVITE request matches a transaction if the Request-URI, To tag, From tag, Call-ID, CSeq, and top Via header field match those of the INVITE request which created the transaction. In this case, the INVITE is a retransmission of the original one that created the transaction. The ACK request matches a transaction if the Request-URI, From tag, Call-ID, CSeq number (not the method), and top Via header field match those of the INVITE request which created the transaction, and the To tag of the ACK matches the To tag of the response sent by the server transaction. Matching is done based on the matching rules defined for each of those header fields. Inclusion of the tag in the To header field in the ACK matching process helps disambiguate ACK for 2xx from ACK for other responses

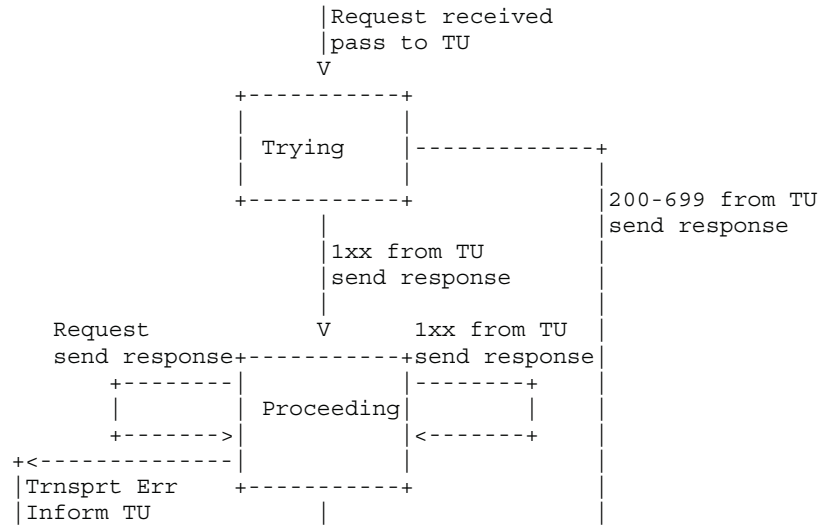




at a proxy, which may have forwarded both responses (This can occur in unusual conditions. Specifically, when a proxy forked a request, and then crashes, the responses may be delivered to another proxy, which might end up forwarding multiple responses upstream). An ACK request that matches an INVITE transaction matched by a previous ACK is considered a retransmission of that previous ACK.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 763
Keysight, EX-1045
Page 763






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 764
Keysight, EX-1045
Page 764

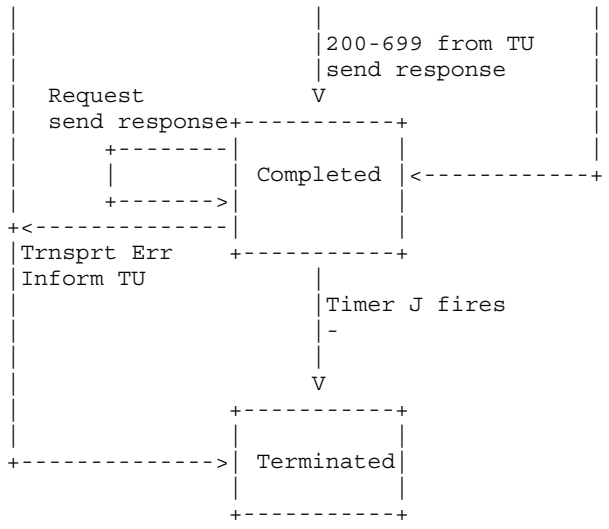



Figure 8: non-INVITE server transaction

For all other request methods, a request is matched to a transaction if the Request-URI, To tag, From tag, Call-ID, CSeq (including the method), and top Via header field match those of the request that created the transaction. Matching is done based on the matching





rules defined for each of those header fields. When a non-INVITE request matches an existing transaction, it is a retransmission of the request that created that transaction.

Because the matching rules include the Request-URI, the server cannot match a response to a transaction. When the TU passes a response to the server transaction, it must pass it to the specific server transaction for which the response is targeted.

### 17.2.4 Handling Transport Errors

When the server transaction sends a response to the transport layer to be sent, the following procedures are followed if the transport layer indicates a failure.

First, the procedures in [4] are followed, which attempt to deliver the response to a backup. If those should all fail, based on the definition of failure in [4], the server transaction SHOULD inform the TU that a failure has occurred, and SHOULD transition to the terminated state.

## 18 Transport

The transport layer is responsible for the actual transmission of requests and responses over network transports. This includes determination of the connection to use for a request or response in the case of connection-oriented transports.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 765
Keysight, EX-1045
Page 765

The transport layer is responsible for managing persistent connections for transport protocols like TCP and SCTP, or TLS over those, including ones opened to the transport layer. This includes connections opened by the client or server transports, so that connections are shared between client and server transport functions. These connections are indexed by the tuple formed from the address, port, and transport protocol at the far end of the connection. When a connection is opened by the transport layer, this index is set to the destination IP, port and transport. When the connection is accepted by the transport layer, this index is set to the source IP address, port number, and transport. Note that, because the source port is often ephemeral, but it cannot be known whether it is ephemeral or selected through procedures in [4], connections accepted by the transport layer will frequently not be reused. The result is that two proxies in a "peering" relationship using a connection-oriented transport frequently will have two connections in use, one for transactions initiated in each direction.





It is RECOMMENDED that connections be kept open for some implementation-defined duration after the last message was sent or received over that connection. This duration SHOULD at least equal the longest amount of time the element would need in order to bring a transaction from instantiation to the terminated state. This is to make it likely that transactions are completed over the same connection on which they are initiated (for example, request, response, and in the case of INVITE, ACK for non-2xx responses). This usually means at least 64*T1 (see Section 17.1.1.1 for a definition of T1). However, it could be larger in an element that has a TU using a large value for timer C (bullet 11 of Section 16.6), for example.

All SIP elements MUST implement UDP and TCP. SIP elements MAY implement other protocols.

> Making TCP mandatory for the UA is a substantial change from RFC 2543. It has arisen out of the need to handle larger messages, which MUST use TCP, as discussed below. Thus, even if an element never sends large messages, it may receive one and needs to be able to handle them.

## 18.1 Clients

### 18.1.1 Sending Requests

The client side of the transport layer is responsible for sending the request and receiving responses. The user of the transport layer passes the client transport the request, an IP address, port, transport, and possibly TTL for multicast destinations.

If a request is within 200 bytes of the path MTU, or if it is larger than 1300 bytes and the path MTU is unknown, the request MUST be sent using an RFC 2914 [43] congestion controlled transport protocol, such

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 766
Keysight, EX-1045
Page 766

as TCP. If this causes a change in the transport protocol from the one indicated in the top Via, the value in the top Via MUST be changed. This prevents fragmentation of messages over UDP and provides congestion control for larger messages. However, implementations MUST be able to handle messages up to the maximum datagram packet size. For UDP, this size is 65,535 bytes, including IP and UDP headers.

The 200 byte "buffer" between the message size and the MTU accommodates the fact that the response in SIP can be larger than the request. This happens due to the addition of Record-Route header field values to the responses to INVITE, for example. With the extra buffer, the response can be about 170 bytes larger than the request, and still not be fragmented on IPv4 (about 30 bytes





is consumed by IP/UDP, assuming no IPSec). 1300 is chosen when path MTU is not known, based on the assumption of a 1500 byte Ethernet MTU.

If an element sends a request over TCP because of these message size constraints, and that request would have otherwise been sent over UDP, if the attempt to establish the connection generates either an ICMP Protocol Not Supported, or results in a TCP reset, the element SHOULD retry the request, using UDP. This is only to provide backwards compatibility with RFC 2543 compliant implementations that do not support TCP. It is anticipated that this behavior will be deprecated in a future revision of this specification.

A client that sends a request to a multicast address MUST add the "maddr" parameter to its Via header field value containing the destination multicast address, and for IPv4, SHOULD add the "ttl" parameter with a value of 1. Usage of IPv6 multicast is not defined in this specification, and will be a subject of future standardization when the need arises.

These rules result in a purposeful limitation of multicast in SIP. Its primary function is to provide a "single-hop-discovery-like" service, delivering a request to a group of homogeneous servers, where it is only required to process the response from any one of them. This functionality is most useful for registrations. In fact, based on the transaction processing rules in Section 17.1.3, the client transaction will accept the first response, and view any others as retransmissions because they all contain the same Via branch identifier.

Before a request is sent, the client transport MUST insert a value of the "sent-by" field into the Via header field. This field contains an IP address or host name, and port. The usage of an FQDN is RECOMMENDED. This field is used for sending responses under certain conditions, described below. If the port is absent, the default value depends on the transport. It is 5060 for UDP, TCP and SCTP, 5061 for TLS.

For reliable transports, the response is normally sent on the connection on which the request was received. Therefore, the client transport MUST be prepared to receive the response on the same

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 767
Keysight, EX-1045
Page 767

connection used to send the request. Under error conditions, the server may attempt to open a new connection to send the response. To handle this case, the transport layer MUST also be prepared to receive an incoming connection on the source IP address from which the request was sent and port number in the "sent-by" field. It also



MUST be prepared to receive incoming connections on any address and port that would be selected by a server based on the procedures described in Section 5 of [4].

For unreliable unicast transports, the client transport MUST be prepared to receive responses on the source IP address from which the request is sent (as responses are sent back to the source address) and the port number in the "sent-by" field. Furthermore, as with reliable transports, in certain cases the response will be sent elsewhere. The client MUST be prepared to receive responses on any address and port that would be selected by a server based on the procedures described in Section 5 of [4].

For multicast, the client transport MUST be prepared to receive responses on the same multicast group and port to which the request is sent (that is, it needs to be a member of the multicast group it sent the request to.)

If a request is destined to an IP address, port, and transport to which an existing connection is open, it is RECOMMENDED that this connection be used to send the request, but another connection MAY be opened and used.

If a request is sent using multicast, it is sent to the group address, port, and TTL provided by the transport user. If a request is sent using unicast unreliable transports, it is sent to the IP address and port provided by the transport user.

### 18.1.2 Receiving Responses

When a response is received, the client transport examines the top Via header field value. If the value of the "sent-by" parameter in that header field value does not correspond to a value that the client transport is configured to insert into requests, the response MUST be silently discarded.

If there are any client transactions in existence, the client transport uses the matching procedures of Section 17.1.3 to attempt to match the response to an existing transaction. If there is a match, the response MUST be passed to that transaction. Otherwise, the response MUST be passed to the core (whether it be stateless proxy, stateful proxy, or UA) for further processing. Handling of these "stray" responses is dependent on the core (a proxy will forward them, while a UA will discard, for example).

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 768
Keysight, EX-1045
Page 768

18.2 Servers

18.2.1 Receiving Requests

A server SHOULD be prepared to receive requests on any IP address, port and transport combination that can be the result of a DNS lookup on a SIP or SIPS URI [4] that is handed out for the purposes of communicating with that server. In this context, "handing out" includes placing a URI in a Contact header field in a REGISTER request or a redirect response, or in a Record-Route header field in a request or response. A URI can also be "handed out" by placing it on a web page or business card. It is also RECOMMENDED that a server listen for requests on the default SIP ports (5060 for TCP and UDP, 5061 for TLS over TCP) on all public interfaces. The typical exception would be private networks, or when multiple server instances are running on the same host. For any port and interface that a server listens on for UDP, it MUST listen on that same port and interface for TCP. This is because a message may need to be sent using TCP, rather than UDP, if it is too large. As a result, the converse is not true. A server need not listen for UDP on a particular address and port just because it is listening on that same address and port for TCP. There may, of course, be other reasons why a server needs to listen for UDP on a particular address and port.

When the server transport receives a request over any transport, it MUST examine the value of the "sent-by" parameter in the top Via header field value. If the host portion of the "sent-by" parameter contains a domain name, or if it contains an IP address that differs from the packet source address, the server MUST add a "received" parameter to that Via header field value. This parameter MUST contain the source address from which the packet was received. This is to assist the server transport layer in sending the response, since it must be sent to the source IP address from which the request came.

Consider a request received by the server transport which looks like, in part:

```
INVITE sip:bob@Biloxi.com SIP/2.0
Via: SIP/2.0/UDP bobspc.biloxi.com:5060
```

The request is received with a source IP address of 192.0.2.4. Before passing the request up, the transport adds a "received" parameter, so that the request would look like, in part:

```
INVITE sip:bob@Biloxi.com SIP/2.0
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;received=192.0.2.4
```



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 769
Keysight, EX-1045
Page 769

Next, the server transport attempts to match the request to a server transaction. It does so using the matching rules described in Section 17.2.3. If a matching server transaction is found, the request is passed to that transaction for processing. If no match is found, the request is passed to the core, which may decide to construct a new server transaction for that request. Note that when a UAS core sends a 2xx response to INVITE, the server transaction is destroyed. This means that when the ACK arrives, there will be no matching server transaction, and based on this rule, the ACK is passed to the UAS core, where it is processed.

18.2.2 Sending Responses

The server transport uses the value of the top Via header field in order to determine where to send a response. It MUST follow the following process:

- If the "sent-protocol" is a reliable transport protocol such as TCP or SCTP, or TLS over those, the response MUST be sent using the existing connection to the source of the original request that created the transaction, if that connection is still open. This requires the server transport to maintain an association between server transactions and transport connections. If that connection is no longer open, the server SHOULD open a connection to the IP address in the "received" parameter, if present, using the port in the "sent-by" value, or the default port for that transport, if no port is specified. If that connection attempt fails, the server SHOULD use the procedures in [4] for servers in order to determine the IP address and port to open the connection and send the response to.

- Otherwise, if the Via header field value contains a "maddr" parameter, the response MUST be forwarded to the address listed there, using the port indicated in "sent-by", or port 5060 if none is present. If the address is a multicast address, the response SHOULD be sent using the TTL indicated in the "ttl" parameter, or with a TTL of 1 if that parameter is not present.

- Otherwise (for unreliable unicast transports), if the top Via has a "received" parameter, the response MUST be sent to the address in the "received" parameter, using the port indicated in the "sent-by" value, or using port 5060 if none is specified explicitly. If this fails, for example, elicits an ICMP "port unreachable" response, the procedures of Section 5 of [4] SHOULD be used to determine where to send the response.



- Otherwise, if it is not receiver-tagged, the response MUST be sent to the address indicated by the "sent-by" value, using the procedures in Section 5 of [4].

18.3 Framing

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 770
Keysight, EX-1045
Page 770

In the case of message-oriented transports (such as UDP), if the message has a Content-Length header field, the message body is assumed to contain that many bytes. If there are additional bytes in the transport packet beyond the end of the body, they MUST be discarded. If the transport packet ends before the end of the message body, this is considered an error. If the message is a response, it MUST be discarded. If the message is a request, the element SHOULD generate a 400 (Bad Request) response. If the message has no Content-Length header field, the message body is assumed to end at the end of the transport packet.

In the case of stream-oriented transports such as TCP, the Content-Length header field indicates the size of the body. The Content-Length header field MUST be used with stream oriented transports.

### 18.4 Error Handling

Error handling is independent of whether the message was a request or response.

If the transport user asks for a message to be sent over an unreliable transport, and the result is an ICMP error, the behavior depends on the type of ICMP error. Host, network, port or protocol unreachable errors, or parameter problem errors SHOULD cause the transport layer to inform the transport user of a failure in sending. Source quench and TTL exceeded ICMP errors SHOULD be ignored.

If the transport user asks for a request to be sent over a reliable transport, and the result is a connection failure, the transport layer SHOULD inform the transport user of a failure in sending.

## 19 Common Message Components

There are certain components of SIP messages that appear in various places within SIP messages (and sometimes, outside of them) that merit separate discussion.





### 19.1 SIP and SIPS Uniform Resource Indicators

A SIP or SIPS URI identifies a communications resource. Like all URIs, SIP and SIPS URIs may be placed in web pages, email messages, or printed literature. They contain sufficient information to initiate and maintain a communication session with the resource.

Examples of communications resources include the following:

- a user of an online service
- an appearance on a multi-line phone

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 771
Keysight, EX-1045
Page 771

- a mailbox on a messaging system
- a PSTN number at a gateway service
- a group (such as "sales" or "helpdesk") in an organization

A SIPS URI specifies that the resource be contacted securely. This means, in particular, that TLS is to be used between the UAC and the domain that owns the URI. From there, secure communications are used to reach the user, where the specific security mechanism depends on the policy of the domain. Any resource described by a SIP URI can be "upgraded" to a SIPS URI by just changing the scheme, if it is desired to communicate with that resource securely.

### 19.1.1 SIP and SIPS URI Components

The "sip:" and "sips:" schemes follow the guidelines in RFC 2396 [5]. They use a form similar to the mailto URL, allowing the specification of SIP request-header fields and the SIP message-body. This makes it possible to specify the subject, media type, or urgency of sessions initiated by using a URI on a web page or in an email message. The formal syntax for a SIP or SIPS URI is presented in Section 25. Its general form, in the case of a SIP URI, is:

```
sip:user:password@host:port;uri-parameters?headers
```

The format for a SIPS URI is the same, except that the scheme is "sips" instead of sip. These tokens, and some of the tokens in their expansions, have the following meanings:

user: The identifier of a particular resource at the host being addressed. The term "host" in this context frequently refers to a domain. The "userinfo" of a URI consists of this user field, the password field, and the @ sign following them. The userinfo part of a URI is optional and MAY be absent when the





destination host does not have a notion of users or when the host itself is the resource being identified. If the @ sign is present in a SIP or SIPS URI, the user field MUST NOT be empty.

If the host being addressed can process telephone numbers, for instance, an Internet telephony gateway, a telephone-subscriber field defined in RFC 2806 [9] MAY be used to populate the user field. There are special escaping rules for encoding telephone-subscriber fields in SIP and SIPS URIs described in Section 19.1.2.

password: A password associated with the user. While the SIP and SIPS URI syntax allows this field to be present, its use is NOT RECOMMENDED, because the passing of authentication information in clear text (such as URIs) has proven to be a security risk in almost every case where it has been used. For instance, transporting a PIN number in this field exposes the PIN.

Note that the password field is just an extension of the user portion. Implementations not wishing to give special

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

significance to the password portion of the field MAY simply treat "user:password" as a single string.

host: The host providing the SIP resource. The host part contains either a fully-qualified domain name or numeric IPv4 or IPv6 address. Using the fully-qualified domain name form is RECOMMENDED whenever possible.

port: The port number where the request is to be sent.

URI parameters: Parameters affecting a request constructed from the URI.

URI parameters are added after the hostport component and are separated by semi-colons.

URI parameters take the form:

parameter-name "=" parameter-value

Even though an arbitrary number of URI parameters may be included in a URI, any given parameter-name MUST NOT appear more than once.

This extensible mechanism includes the transport, maddr, ttl, user, method and lr parameters.





The transport parameter determines the transport mechanism to be used for sending SIP messages, as specified in [4]. SIP can use any network transport protocol. Parameter names are defined for UDP (RFC 768 [14]), TCP (RFC 761 [15]), and SCTP (RFC 2960 [16]). For a SIPS URI, the transport parameter MUST indicate a reliable transport.

The maddr parameter indicates the server address to be contacted for this user, overriding any address derived from the host field. When an maddr parameter is present, the port and transport components of the URI apply to the address indicated in the maddr parameter value. [4] describes the proper interpretation of the transport, maddr, and hostport in order to obtain the destination address, port, and transport for sending a request.

The maddr field has been used as a simple form of loose source routing. It allows a URI to specify a proxy that must be traversed en-route to the destination. Continuing to use the maddr parameter this way is strongly discouraged (the mechanisms that enable it are deprecated). Implementations should instead use the Route mechanism described in this document, establishing a pre-existing route set if necessary (see Section 8.1.1.1). This provides a full URI to describe the node to be traversed.

The ttl parameter determines the time-to-live value of the UDP

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

multicast packet and MUST only be used if maddr is a multicast address and the transport protocol is UDP. For example, to specify a call to alice@atlanta.com using multicast to 239.255.255.1 with a ttl of 15, the following URI would be used:

sip:alice@atlanta.com;maddr=239.255.255.1;ttl=15

The set of valid telephone-subscriber strings is a subset of valid user strings. The user URI parameter exists to distinguish telephone numbers from user names that happen to look like telephone numbers. If the user string contains a telephone number formatted as a telephone-subscriber, the user parameter value "phone" SHOULD be present. Even without this parameter, recipients of SIP and SIPS URIs MAY interpret the pre-@ part as a telephone number if local restrictions on the name space for user name allow it.

The method of the SIP request constructed from the URI can be specified with the method parameter.





The lr parameter, when present, indicates that the element responsible for this resource implements the routing mechanisms specified in this document. This parameter will be used in the URIs proxies place into Record-Route header field values, and may appear in the URIs in a pre-existing route set.

This parameter is used to achieve backwards compatibility with systems implementing the strict-routing mechanisms of RFC 2543 and the rfc2543bis drafts up to bis-05. An element preparing to send a request based on a URI not containing this parameter can assume the receiving element implements strict-routing and reformat the message to preserve the information in the Request-URI.

Since the uri-parameter mechanism is extensible, SIP elements MUST silently ignore any uri-parameters that they do not understand.

Headers: Header fields to be included in a request constructed from the URI.

Headers fields in the SIP request can be specified with the "?" mechanism within a URI. The header names and values are encoded in ampersand separated hname = hvalue pairs. The special hname "body" indicates that the associated hvalue is the message-body of the SIP request.

Table 1 summarizes the use of SIP and SIPS URI components based on the context in which the URI appears. The external column describes URIs appearing anywhere outside of a SIP message, for instance on a web page or business card. Entries marked "m" are mandatory, those marked "o" are optional, and those marked "-" are not allowed. Elements processing URIs SHOULD ignore any disallowed components if they are present. The second column indicates the default value of

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 774
Keysight, EX-1045
Page 774

an optional element if it is not present. "--" indicates that the element is either not optional, or has no default value.

URIs in Contact header fields have different restrictions depending on the context in which the header field appears. One set applies to messages that establish and maintain dialogs (INVITE and its 200 (OK) response). The other applies to registration and redirection messages (REGISTER, its 200 (OK) response, and 3xx class responses to any method).





19.1.2 Character Escaping Requirements

| | default | Req.-URI | To | From | reg./redir. Contact | dialog Contact/ R-R/Route | external |
|---|---|---|---|---|---|---|---|
| user | -- | o | o | o | o | o | o |
| password | -- | o | o | o | o | o | o |
| host | -- | m | m | m | m | m | m |
| port | (1) | o | - | - | o | o | o |
| user-param | ip | o | o | o | o | o | o |
| method | INVITE | - | - | - | - | - | o |
| maddr-param | -- | o | - | - | o | o | o |
| ttl-param | 1 | o | - | - | o | - | o |
| transp.-param | (2) | o | - | - | o | o | o |
| lr-param | -- | o | - | - | - | o | o |
| other-param | -- | o | o | o | o | o | o |
| headers | -- | - | - | - | o | - | o |

(1): The default port value is transport and scheme dependent. The default is 5060 for sip: using UDP, TCP, or SCTP. The default is 5061 for sip: using TLS over TCP and sips: over TCP.

(2): The default transport is scheme dependent. For sip:, it is UDP. For sips:, it is TCP.

Table 1: Use and default values of URI components for SIP header field values, Request-URI and references

SIP follows the requirements and guidelines of RFC 2396 [5] when defining the set of characters that must be escaped in a SIP URI, and uses its ""%" HEX HEX" mechanism for escaping. From RFC 2396 [5]:

> The set of characters actually reserved within any given URI component is defined by that component. In general, a character is reserved if the semantics of the URI changes if the character is replaced with its escaped US-ASCII encoding [5]. Excluded US-ASCII characters (RFC 2396 [5]), such as space and control characters and characters used as URI delimiters, also MUST be escaped. URIs MUST NOT contain unescaped space and control characters.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 775
Keysight, EX-1045
Page 775

For each component, the set of valid BNF expansions defines exactly which characters may appear unescaped. All other characters MUST be escaped.

For example, "@" is not in the set of characters in the user component, so the user "j@s0n" must have at least the @ sign encoded, as in "j%40s0n".



Expanding the hname and hvalue tokens in Section 25 show that all URI reserved characters in header field names and values MUST be escaped.

The telephone-subscriber subset of the user component has special escaping considerations. The set of characters not reserved in the RFC 2806 [9] description of telephone-subscriber contains a number of characters in various syntax elements that need to be escaped when used in SIP URIs. Any characters occurring in a telephone-subscriber that do not appear in an expansion of the BNF for the user rule MUST be escaped.

Note that character escaping is not allowed in the host component of a SIP or SIPS URI (the % character is not valid in its expansion). This is likely to change in the future as requirements for Internationalized Domain Names are finalized. Current implementations MUST NOT attempt to improve robustness by treating received escaped characters in the host component as literally equivalent to their unescaped counterpart. The behavior required to meet the requirements of IDN may be significantly different.

### 19.1.3 Example SIP and SIPS URIs

```
sip:alice@atlanta.com
sip:alice:secretword@atlanta.com;transport=tcp
sips:alice@atlanta.com?subject=project%20x&priority=urgent
sip:+1-212-555-1212:1234@gateway.com;user=phone
sips:1212@gateway.com
sip:alice@192.0.2.4
sip:atlanta.com;method=REGISTER?to=alice%40atlanta.com
sip:alice;day=tuesday@atlanta.com
```

The last sample URI above has a user field value of "alice;day=tuesday". The escaping rules defined above allow a semicolon to appear unescaped in this field. For the purposes of this protocol, the field is opaque. The structure of that value is only useful to the SIP element responsible for the resource.

### 19.1.4 URI Comparison

Some operations in this specification require determining whether two SIP or SIPS URIs are equivalent. In this specification, registrars need to compare bindings in Contact URIs in REGISTER requests (see Section 10.3.). SIP and SIPS URIs are compared for equality according to the following rules:

- A SIP and SIPS URI are never equivalent.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

- Comparison of the userinfo of SIP and SIPS URIs is case-sensitive. This includes userinfo containing passwords or formatted as telephone-subscribers. Comparison of all other components of the URI is case-insensitive unless explicitly defined otherwise.

- The ordering of parameters and header fields is not significant in comparing SIP and SIPS URIs.

- Characters other than those in the "reserved" set (see RFC 2396 [5]) are equivalent to their ""%" HEX HEX" encoding.

- An IP address that is the result of a DNS lookup of a host name does not match that host name.

- For two URIs to be equal, the user, password, host, and port components must match.

  A URI omitting the user component will not match a URI that includes one. A URI omitting the password component will not match a URI that includes one.

  A URI omitting any component with a default value will not match a URI explicitly containing that component with its default value. For instance, a URI omitting the optional port component will not match a URI explicitly declaring port 5060. The same is true for the transport-parameter, ttl-parameter, user-parameter, and method components.

  Defining sip:user@host to not be equivalent to sip:user@host:5060 is a change from RFC 2543. When deriving addresses from URIs, equivalent addresses are expected from equivalent URIs. The URI sip:user@host:5060 will always resolve to port 5060. The URI sip:user@host may resolve to other ports through the DNS SRV mechanisms detailed in [4].

- URI uri-parameter components are compared as follows:

  - Any uri-parameter appearing in both URIs must match.

  - A user, ttl, or method uri-parameter appearing in only one URI never matches, even if it contains the default value.

  - A URI that includes an maddr parameter will not match a URI that contains no maddr parameter.

  - All other uri-parameters appearing in only one URI are ignored when comparing the URIs.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 777
Keysight, EX-1045
Page 777

- URI header components are never ignored. Any present header component MUST be present in both URIs and match for the URIs to match. The matching rules are defined for each header field in Section 20.

The URIs within each of the following sets are equivalent:

```
sip:%61lice@atlanta.com;transport=TCP
sip:alice@AtLanTa.CoM;Transport=tcp

sip:carol@chicago.com
sip:carol@chicago.com;newparam=5
sip:carol@chicago.com;security=on

sip:biloxi.com;transport=tcp;method=REGISTER?to=sip:bob%40biloxi.com
sip:biloxi.com;method=REGISTER;transport=tcp?to=sip:bob%40biloxi.com

sip:alice@atlanta.com?subject=project%20x&priority=urgent
sip:alice@atlanta.com?priority=urgent&subject=project%20x
```

The URIs within each of the following sets are not equivalent:

```
SIP:ALICE@AtLanTa.CoM;Transport=udp             (different usernames)
sip:alice@AtLanTa.CoM;Transport=UDP

sip:bob@biloxi.com                   (can resolve to different ports)
sip:bob@biloxi.com:5060

sip:bob@biloxi.com              (can resolve to different transports)
sip:bob@biloxi.com;transport=udp

sip:bob@biloxi.com     (can resolve to different port and transports)
sip:bob@biloxi.com:6000;transport=tcp

sip:carol@chicago.com                     (different header component)
sip:carol@chicago.com?Subject=next%20meeting

sip:bob@phone21.boxesbybob.com   (even though that's what
sip:bob@192.0.2.4                 phone21.boxesbybob.com resolves to)
```

Note that equality is not transitive:

- sip:carol@chicago.com and sip:carol@chicago.com;security=on are equivalent
- sip:carol@chicago.com and sip:carol@chicago.com;security=off are equivalent



- sip:carol@chicago.com;security=on and sip:carol@chicago.com;security=off are not equivalent

### 19.1.5 Forming Requests from a URI

An implementation needs to take care when forming requests directly

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 778
Keysight, EX-1045
Page 778

from a URI. URIs from business cards, web pages, and even from sources inside the protocol such as registered contacts may contain inappropriate header fields or body parts.

An implementation MUST include any provided transport, maddr, ttl, or user parameter in the Request-URI of the formed request. If the URI contains a method parameter, its value MUST be used as the method of the request. The method parameter MUST NOT be placed in the Request-URI. Unknown URI parameters MUST be placed in the message's Request-URI.

An implementation SHOULD treat the presence of any headers or body parts in the URI as a desire to include them in the message, and choose to honor the request on a per-component basis.

An implementation SHOULD NOT honor these obviously dangerous header fields: From, Call-ID, CSeq, Via, and Record-Route.

An implementation SHOULD NOT honor any requested Route header field values in order to not be used as an unwitting agent in malicious attacks.

An implementation SHOULD NOT honor requests to include header fields that may cause it to falsely advertise its location or capabilities. These include: Accept, Accept-Encoding, Accept-Language, Allow, Contact (in its dialog usage), Organization, Supported, and User-Agent.

An implementation SHOULD verify the accuracy of any requested descriptive header fields, including: Content-Disposition, Content-Encoding, Content-Language, Content-Length, Content-Type, Date, Mime-Version, and Timestamp.

If the request formed from constructing a message from a given URI is not a valid SIP request, the URI is invalid. An implementation MUST NOT proceed with transmitting the request. It should instead pursue the course of action due an invalid URI in the context it occurs.

> The constructed request can be invalid in many ways. These include, but are not limited to, syntax error in header fields, invalid combinations of URI parameters, or an incorrect description of the message body.





Sending a request formed from a given URI may require capabilities unavailable to the implementation. The URI might indicate use of an unimplemented transport or extension, for example. An implementation SHOULD refuse to send these requests rather than modifying them to match their capabilities. An implementation MUST NOT send a request requiring an extension that it does not support.

> For example, such a request can be formed through the presence of a Require header parameter or a method URI parameter with an unknown or explicitly unsupported value.

### 19.1.6 Relating SIP URIs and tel URLs

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 779
Keysight, EX-1045
Page 779

When a tel URL (RFC 2806 [9]) is converted to a SIP or SIPS URI, the entire telephone-subscriber portion of the tel URL, including any parameters, is placed into the userinfo part of the SIP or SIPS URI.

Thus, tel:+358-555-1234567;postd=pp22 becomes

sip:+358-555-1234567;postd=pp22@foo.com;user=phone

or

sips:+358-555-1234567;postd=pp22@foo.com;user=phone

not

sip:+358-555-1234567@foo.com;postd=pp22;user=phone

or

sips:+358-555-1234567@foo.com;postd=pp22;user=phone

In general, equivalent "tel" URLs converted to SIP or SIPS URIs in this fashion may not produce equivalent SIP or SIPS URIs. The userinfo of SIP and SIPS URIs are compared as a case-sensitive string. Variance in case-insensitive portions of tel URLs and reordering of tel URL parameters does not affect tel URL equivalence, but does affect the equivalence of SIP URIs formed from them.

For example,

tel:+358-555-1234567;postd=pp22
tel:+358-555-1234567;POSTD=PP22

are equivalent, while

sip:+358-555-1234567;postd=pp22@foo.com;user=phone
sip:+358-555-1234567;POSTD=PP22@foo.com;user=phone





are not.

Likewise,

tel:+358-555-1234567;postd=pp22;isub=1411
tel:+358-555-1234567;isub=1411;postd=pp22

are equivalent, while

sip:+358-555-1234567;postd=pp22;isub=1411@foo.com;user=phone
sip:+358-555-1234567;isub=1411;postd=pp22@foo.com;user=phone

are not.

To mitigate this problem, elements constructing telephone-subscriber fields to place in the userinfo part of a SIP or SIPS URI SHOULD fold any case-insensitive portion of telephone-subscriber to lower case, and order the telephone-subscriber parameters lexically by parameter name, excepting isdn-subaddress and post-dial, which occur first and in that order. (All components of a tel URL except for future-

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 780
Keysight, EX-1045
Page 780

extension parameters are defined to be compared case-insensitive.)

Following this suggestion, both

```
tel:+358-555-1234567;postd=pp22
tel:+358-555-1234567;POSTD=PP22
```

become

```
sip:+358-555-1234567;postd=pp22@foo.com;user=phone
```

and both

```
tel:+358-555-1234567;tsp=a.b;phone-context=5
tel:+358-555-1234567;phone-context=5;tsp=a.b
```

become

```
sip:+358-555-1234567;phone-context=5;tsp=a.b@foo.com;user=phone
```

## 19.2 Option Tags

Option tags are unique identifiers used to designate new options (extensions) in SIP. These tags are used in Require (Section 20.32), Proxy-Require (Section 20.29), Supported (Section 20.37) and Unsupported (Section 20.40) header fields. Note that these options appear as parameters in those header fields in an option-tag = token form (see Section 25 for the definition of token).





Option tags are defined in standards track RFCs. This is a change from past practice, and is instituted to ensure continuing multi-vendor interoperability (see discussion in Section 20.32 and Section 20.37). An IANA registry of option tags is used to ensure easy reference.

## 19.3 Tags

The "tag" parameter is used in the To and From header fields of SIP messages. It serves as a general mechanism to identify a dialog, which is the combination of the Call-ID along with two tags, one from each participant in the dialog. When a UA sends a request outside of a dialog, it contains a From tag only, providing "half" of the dialog ID. The dialog is completed from the response(s), each of which contributes the second half in the To header field. The forking of SIP requests means that multiple dialogs can be established from a single request. This also explains the need for the two-sided dialog identifier; without a contribution from the recipients, the originator could not disambiguate the multiple dialogs established from a single request.

When a tag is generated by a UA for insertion into a request or response, it MUST be globally unique and cryptographically random with at least 32 bits of randomness. A property of this selection requirement is that a UA will place a different tag into the From header of an INVITE than it would place into the To header of the response to the same INVITE. This is needed in order for a UA to

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 781
Keysight, EX-1045
Page 781

invite itself to a session, a common case for "hairpinning" of calls in PSTN gateways. Similarly, two INVITEs for different calls will have different From tags, and two responses for different calls will have different To tags.

Besides the requirement for global uniqueness, the algorithm for generating a tag is implementation-specific. Tags are helpful in fault tolerant systems, where a dialog is to be recovered on an alternate server after a failure. A UAS can select the tag in such a way that a backup can recognize a request as part of a dialog on the failed server, and therefore determine that it should attempt to recover the dialog and any other state associated with it.

## 20 Header Fields

The general syntax for header fields is covered in Section 7.3. This section lists the full set of header fields along with notes on syntax, meaning, and usage. Throughout this section, we use [HX.Y] to refer to Section X.Y of the current HTTP/1.1 specification RFC 2616 [8]. Examples of each header field are given.





Information about header fields in relation to methods and proxy processing is summarized in Tables 2 and 3.

The "where" column describes the request and response types in which the header field can be used. Values in this column are:

R: header field may only appear in requests;

r: header field may only appear in responses;

2xx, 4xx, etc.: A numerical value or range indicates response codes with which the header field can be used;

c: header field is copied from the request to the response.

An empty entry in the "where" column indicates that the header field may be present in all requests and responses.

The "proxy" column describes the operations a proxy may perform on a header field:

a: A proxy can add or concatenate the header field if not present.

m: A proxy can modify an existing header field value.

d: A proxy can delete a header field value.

r: A proxy must be able to read the header field, and thus this header field cannot be encrypted.

The next six columns relate to the presence of a header field in a method:

c: Conditional; requirements on the header field depend on the

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 782
Keysight, EX-1045
Page 782

context of the message.

m: The header field is mandatory.

m*: The header field SHOULD be sent, but clients/servers need to be prepared to receive messages without that header field.

o: The header field is optional.

t: The header field SHOULD be sent, but clients/servers need to be prepared to receive messages without that header field.

If a stream-based protocol (such as TCP) is used as a transport, then the header field MUST be sent.





*: The header field is required if the message body is not empty. See Sections 20.14, 20.15 and 7.4 for details.

-: The header field is not applicable.

"Optional" means that an element MAY include the header field in a request or response, and a UA MAY ignore the header field if present in the request or response (The exception to this rule is the Require header field discussed in 20.32). A "mandatory" header field MUST be present in a request, and MUST be understood by the UAS receiving the request. A mandatory response header field MUST be present in the response, and the header field MUST be understood by the UAC processing the response. "Not applicable" means that the header field MUST NOT be present in a request. If one is placed in a request by mistake, it MUST be ignored by the UAS receiving the request. Similarly, a header field labeled "not applicable" for a response means that the UAS MUST NOT place the header field in the response, and the UAC MUST ignore the header field in the response.

A UA SHOULD ignore extension header parameters that are not understood.

A compact form of some common header field names is also defined for use when overall message size is an issue.

The Contact, From, and To header fields contain a URI. If the URI contains a comma, question mark or semicolon, the URI MUST be enclosed in angle brackets (< and >). Any URI parameters are contained within these brackets. If the URI is not enclosed in angle brackets, any semicolon-delimited parameters are header-parameters, not URI parameters.

### 20.1 Accept

The Accept header field follows the syntax defined in [H14.1]. The semantics are also identical, with the exception that if no Accept header field is present, the server SHOULD assume a default value of application/sdp.

An empty Accept header field means that no formats are acceptable.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 783
Keysight, EX-1045
Page 783

Example:

| Header field | where | proxy | ACK | BYE | CAN | INV | OPT | REG |
|---|---|---|---|---|---|---|---|---|
| Accept | R | | - | o | - | o | m* | o |
| Accept | 2xx | | - | - | - | o | m* | o |
| Accept | 415 | | - | c | - | c | c | c |
| Accept-Encoding | R | | - | o | - | o | o | o |
| Accept-Encoding | 2xx | | - | - | - | o | m* | o |
| Accept-Encoding | 415 | | - | c | - | c | c | c |
| Accept-Language | R | | - | o | - | o | o | o |
| Accept-Language | 2xx | | - | - | - | o | m* | o |
| Accept-Language | 415 | | - | c | - | c | c | c |
| Alert-Info | R | ar | - | - | - | o | - | - |
| Alert-Info | 180 | ar | - | - | - | o | - | - |
| Allow | R | | - | o | - | o | o | o |
| Allow | 2xx | | - | o | - | m* | m* | o |
| Allow | r | | - | o | - | o | o | o |
| Allow | 405 | | - | m | - | m | m | m |
| Authentication-Info | 2xx | | - | o | - | o | o | o |
| Authorization | R | | o | o | o | o | o | o |
| Call-ID | c | r | m | m | m | m | m | m |
| Call-Info | | ar | - | - | - | o | o | o |
| Contact | R | | o | - | - | m | o | o |
| Contact | 1xx | | - | - | - | o | - | - |
| Contact | 2xx | | - | - | - | m | o | o |
| Contact | 3xx | d | - | o | - | o | o | o |
| Contact | 485 | | - | o | - | o | o | o |
| Content-Disposition | | | o | o | - | o | o | o |
| Content-Encoding | | | o | o | - | o | o | o |
| Content-Language | | | o | o | - | o | o | o |
| Content-Length | | ar | t | t | t | t | t | t |
| Content-Type | | | * | * | - | * | * | * |
| CSeq | c | r | m | m | m | m | m | m |
| Date | | a | o | o | o | o | o | o |
| Error-Info | 300-699 | a | - | o | o | o | o | o |
| Expires | | | - | - | - | o | - | o |
| From | c | r | m | m | m | m | m | m |
| In-Reply-To | R | | - | - | - | o | - | - |
| Max-Forwards | R | amr | m | m | m | m | m | m |
| Min-Expires | 423 | | - | - | - | - | - | m |
| MIME-Version | | | o | o | - | o | o | o |
| Organization | | ar | - | - | - | o | o | o |

Table 2: Summary of header fields, A--O

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

| Header field | where | proxy | ACK | BYE | CAN | INV | OPT | REG |
|---|---|---|---|---|---|---|---|---|
| Priority | R | ar | - | - | - | o | - | - |
| Proxy-Authenticate | 407 | ar | - | m | - | m | m | m |
| Proxy-Authenticate | 401 | ar | - | o | o | o | o | o |
| Proxy-Authorization | R | dr | o | o | - | o | o | o |
| Proxy-Require | R | ar | - | o | - | o | o | o |
| Record-Route | R | ar | o | o | o | o | o | - |
| Record-Route | 2xx,18x | mr | - | o | o | o | o | - |
| Reply-To | | | - | - | - | o | - | - |
| Require | | ar | - | c | - | c | c | c |
| Retry-After | 404,413,480,486 | | - | o | o | o | o | o |
| | 500,503 | | - | o | o | o | o | o |
| | 600,603 | | - | o | o | o | o | o |
| Route | R | adr | c | c | c | c | c | c |
| Server | r | | - | o | o | o | o | o |
| Subject | R | | - | - | - | o | - | - |
| Supported | R | | - | o | o | m* | o | o |
| Supported | 2xx | | - | o | o | m* | m* | o |
| Timestamp | | | o | o | o | o | o | o |
| To | c(1) | r | m | m | m | m | m | m |
| Unsupported | 420 | | - | m | - | m | m | m |
| User-Agent | | | o | o | o | o | o | o |
| Via | R | amr | m | m | m | m | m | m |
| Via | rc | dr | m | m | m | m | m | m |
| Warning | r | | - | o | o | o | o | o |
| WWW-Authenticate | 401 | ar | - | m | - | m | m | m |
| WWW-Authenticate | 407 | ar | - | o | - | o | o | o |

Table 3: Summary of header fields, P--Z; (1): copied with possible addition of tag

```
   Accept: application/sdp;level=1, application/x-private, text/html
```

### 20.2 Accept-Encoding

The Accept-Encoding header field is similar to Accept, but restricts the content-codings [H3.5] that are acceptable in the response. See [H14.3]. The semantics in SIP are identical to those defined in [H14.3].

An empty Accept-Encoding header field is permissible. It is equivalent to Accept-Encoding: identity, that is, only the identity encoding, meaning no encoding, is permissible.

If no Accept-Encoding header field is present, the server SHOULD assume a default value of identity.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 785
Keysight, EX-1045
Page 785

This differs slightly from the HTTP definition, which indicates that when not present, any encoding can be used, but the identity encoding is preferred.

Example:

Accept-Encoding: gzip

20.3 Accept-Language

The Accept-Language header field is used in requests to indicate the preferred languages for reason phrases, session descriptions, or status responses carried as message bodies in the response. If no Accept-Language header field is present, the server SHOULD assume all languages are acceptable to the client.

The Accept-Language header field follows the syntax defined in [H14.4]. The rules for ordering the languages based on the "q" parameter apply to SIP as well.

Example:

Accept-Language: da, en-gb;q=0.8, en;q=0.7

20.4 Alert-Info

When present in an INVITE request, the Alert-Info header field specifies an alternative ring tone to the UAS. When present in a 180 (Ringing) response, the Alert-Info header field specifies an alternative ringback tone to the UAC. A typical usage is for a proxy to insert this header field to provide a distinctive ring feature.

The Alert-Info header field can introduce security risks. These risks and the ways to handle them are discussed in Section 20.9, which discusses the Call-Info header field since the risks are identical.

In addition, a user SHOULD be able to disable this feature selectively.

This helps prevent disruptions that could result from the use of this header field by untrusted elements.

Example:

Alert-Info: <http://www.example.com/sounds/moo.wav>



20.5 Allow

The Allow header field lists the set of methods supported by the UA generating the message.

All methods, including ACK and CANCEL, understood by the UA MUST be

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 786
Keysight, EX-1045
Page 786

included in the list of methods in the Allow header field, when present. The absence of an Allow header field MUST NOT be interpreted to mean that the UA sending the message supports no methods. Rather, it implies that the UA is not providing any information on what methods it supports.

Supplying an Allow header field in responses to methods other than OPTIONS reduces the number of messages needed.

Example:

```
Allow: INVITE, ACK, OPTIONS, CANCEL, BYE
```

### 20.6 Authentication-Info

The Authentication-Info header field provides for mutual authentication with HTTP Digest. A UAS MAY include this header field in a 2xx response to a request that was successfully authenticated using digest based on the Authorization header field.

Syntax and semantics follow those specified in RFC 2617 [17].

Example:

```
Authentication-Info: nextnonce="47364c23432d2e131a5fb210812c"
```

### 20.7 Authorization

The Authorization header field contains authentication credentials of a UA. Section 22.2 overviews the use of the Authorization header field, and Section 22.4 describes the syntax and semantics when used with HTTP authentication.

This header field, along with Proxy-Authorization, breaks the general rules about multiple header field values. Although not a comma-separated list, this header field name may be present multiple times, and MUST NOT be combined into a single header line using the usual rules described in Section 7.3.





In the example below, there are no quotes around the Digest parameter:

```
Authorization: Digest username="Alice", realm="atlanta.com",
 nonce="84a4cc6f3082121f32b42a2187831a9e",
 response="7587245234b3434cc3412213e5f113a5432"
```

### 20.8 Call-ID

The Call-ID header field uniquely identifies a particular invitation or all registrations of a particular client. A single multimedia conference can give rise to several calls with different Call-IDs, for example, if a user invites a single individual several times to

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 787
Keysight, EX-1045
Page 787

the same (long-running) conference. Call-IDs are case-sensitive and are simply compared byte-by-byte.

The compact form of the Call-ID header field is i.

Examples:

```
Call-ID: f81d4fae-7dec-11d0-a765-00a0c91e6bf6@biloxi.com
i:f81d4fae-7dec-11d0-a765-00a0c91e6bf6@192.0.2.4
```

### 20.9 Call-Info

The Call-Info header field provides additional information about the caller or callee, depending on whether it is found in a request or response. The purpose of the URI is described by the "purpose" parameter. The "icon" parameter designates an image suitable as an iconic representation of the caller or callee. The "info" parameter describes the caller or callee in general, for example, through a web page. The "card" parameter provides a business card, for example, in vCard [36] or LDIF [37] formats. Additional tokens can be registered using IANA and the procedures in Section 27.

Use of the Call-Info header field can pose a security risk. If a callee fetches the URIs provided by a malicious caller, the callee may be at risk for displaying inappropriate or offensive content, dangerous or illegal content, and so on. Therefore, it is RECOMMENDED that a UA only render the information in the Call-Info header field if it can verify the authenticity of the element that originated the header field and trusts that element. This need not be the peer UA; a proxy can insert this header field into requests.

Example:

```
Call-Info: <http://wwww.example.com/alice/photo.jpg> ;purpose=icon,
  <http://www.example.com/alice/> ;purpose=info
```





### 20.10 Contact

A Contact header field value provides a URI whose meaning depends on the type of request or response it is in.

A Contact header field value can contain a display name, a URI with URI parameters, and header parameters.

This document defines the Contact parameters "q" and "expires". These parameters are only used when the Contact is present in a REGISTER request or response, or in a 3xx response. Additional parameters may be defined in other specifications.

When the header field value contains a display name, the URI including all URI parameters is enclosed in "<" and ">". If no "<" and ">" are present, all parameters after the URI are header parameters, not URI parameters. The display name can be tokens, or a quoted string, if a larger character set is desired.

Even if the "display-name" is empty, the "name-addr" form MUST be

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

used if the "addr-spec" contains a comma, semicolon, or question mark. There may or may not be LWS between the display-name and the "<".

These rules for parsing a display name, URI and URI parameters, and header parameters also apply for the header fields To and From.

The Contact header field has a role similar to the Location header field in HTTP. However, the HTTP header field only allows one address, unquoted. Since URIs can contain commas and semicolons as reserved characters, they can be mistaken for header or parameter delimiters, respectively.

The compact form of the Contact header field is m (for "moved").

Examples:

```
Contact: "Mr. Watson" <sip:watson@worcester.bell-telephone.com>
   ;q=0.7; expires=3600,
   "Mr. Watson" <mailto:watson@bell-telephone.com> ;q=0.1
m: <sips:bob@192.0.2.4>;expires=60
```





### 20.11 Content-Disposition

The Content-Disposition header field describes how the message body or, for multipart messages, a message body part is to be interpreted by the UAC or UAS. This SIP header field extends the MIME Content-Type (RFC 2183 [18]).

Several new "disposition-types" of the Content-Disposition header are defined by SIP. The value "session" indicates that the body part describes a session, for either calls or early (pre-call) media. The value "render" indicates that the body part should be displayed or otherwise rendered to the user. Note that the value "render" is used rather than "inline" to avoid the connotation that the MIME body is displayed as a part of the rendering of the entire message (since the MIME bodies of SIP messages oftentimes are not displayed to users). For backward-compatibility, if the Content-Disposition header field is missing, the server SHOULD assume bodies of Content-Type application/sdp are the disposition "session", while other content types are "render".

The disposition type "icon" indicates that the body part contains an image suitable as an iconic representation of the caller or callee that could be rendered informationally by a user agent when a message has been received, or persistently while a dialog takes place. The value "alert" indicates that the body part contains information, such as an audio clip, that should be rendered by the user agent in an attempt to alert the user to the receipt of a request, generally a

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

request that initiates a dialog; this alerting body could for example be rendered as a ring tone for a phone call after a 180 Ringing provisional response has been sent.

Any MIME body with a "disposition-type" that renders content to the user should only be processed when a message has been properly authenticated.

The handling parameter, handling-param, describes how the UAS should react if it receives a message body whose content type or disposition type it does not understand. The parameter has defined values of "optional" and "required". If the handling parameter is missing, the value "required" SHOULD be assumed. The handling parameter is described in RFC 3204 [19].

If this header field is missing, the MIME type determines the default content disposition. If there is none, "render" is assumed.

Example:

```
Content-Disposition: session
```





### 20.12 Content-Encoding

The Content-Encoding header field is used as a modifier to the "media-type". When present, its value indicates what additional content codings have been applied to the entity-body, and thus what decoding mechanisms MUST be applied in order to obtain the media-type referenced by the Content-Type header field. Content-Encoding is primarily used to allow a body to be compressed without losing the identity of its underlying media type.

If multiple encodings have been applied to an entity-body, the content codings MUST be listed in the order in which they were applied.

All content-coding values are case-insensitive. IANA acts as a registry for content-coding value tokens. See [H3.5] for a definition of the syntax for content-coding.

Clients MAY apply content encodings to the body in requests. A server MAY apply content encodings to the bodies in responses. The server MUST only use encodings listed in the Accept-Encoding header field in the request.

The compact form of the Content-Encoding header field is e. Examples:

```
Content-Encoding: gzip
e: tar
```

### 20.13 Content-Language

See [H14.12]. Example:

```
Content-Language: fr
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

20.14 Content-Length

The Content-Length header field indicates the size of the message-body, in decimal number of octets, sent to the recipient. Applications SHOULD use this field to indicate the size of the message-body to be transferred, regardless of the media type of the entity. If a stream-based protocol (such as TCP) is used as transport, the header field MUST be used.

The size of the message-body does not include the CRLF separating header fields and body. Any Content-Length greater than or equal to zero is a valid value. If no body is present in a message, then the Content-Length header field value MUST be set to zero.





The ability to omit Content-Length simplifies the creation of cgi-like scripts that dynamically generate responses.

The compact form of the header field is l.

Examples:

```
Content-Length: 349
l: 173
```

20.15 Content-Type

The Content-Type header field indicates the media type of the message-body sent to the recipient. The "media-type" element is defined in [H3.7]. The Content-Type header field MUST be present if the body is not empty. If the body is empty, and a Content-Type header field is present, it indicates that the body of the specific type has zero length (for example, an empty audio file).

The compact form of the header field is c.

Examples:

```
Content-Type: application/sdp
c: text/html; charset=ISO-8859-4
```

20.16 CSeq

A CSeq header field in a request contains a single decimal sequence number and the request method. The sequence number MUST be expressible as a 32-bit unsigned integer. The method part of CSeq is case-sensitive. The CSeq header field serves to order transactions within a dialog, to provide a means to uniquely identify transactions, and to differentiate between new requests and request retransmissions. Two CSeq header fields are considered equal if the sequence number and the request method are identical. Example:

```
CSeq: 4711 INVITE
```

20.17 Date

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

The Date header field contains the date and time. Unlike HTTP/1.1, SIP only supports the most recent RFC 1123 [20] format for dates. As in [H3.3], SIP restricts the time zone in SIP-date to "GMT", while RFC 1123 allows any time zone. An RFC 1123 date is case-sensitive.

The Date header field reflects the time when the request or response is first sent.



The Date header field can be used by simple end systems without a battery-backed clock to acquire a notion of current time. However, in its GMT form, it requires clients to know their offset from GMT.

Example:

```
Date: Sat, 13 Nov 2010 23:29:00 GMT
```

### 20.18 Error-Info

The Error-Info header field provides a pointer to additional information about the error status response.

SIP UACs have user interface capabilities ranging from pop-up windows and audio on PC softclients to audio-only on "black" phones or endpoints connected via gateways. Rather than forcing a server generating an error to choose between sending an error status code with a detailed reason phrase and playing an audio recording, the Error-Info header field allows both to be sent. The UAC then has the choice of which error indicator to render to the caller.

A UAC MAY treat a SIP or SIPS URI in an Error-Info header field as if it were a Contact in a redirect and generate a new INVITE, resulting in a recorded announcement session being established. A non-SIP URI MAY be rendered to the user.

Examples:

```
SIP/2.0 404 The number you have dialed is not in service
Error-Info: <sip:not-in-service-recording@atlanta.com>
```

### 20.19 Expires

The Expires header field gives the relative time after which the message (or content) expires.

The precise meaning of this is method dependent.

The expiration time in an INVITE does not affect the duration of the actual session that may result from the invitation. Session description protocols may offer the ability to express time limits on the session duration, however.

The value of this field is an integral number of seconds (in decimal) between 0 and (2**32)-1, measured from the receipt of the request.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

Example:

Expires: 5

20.20 From

The From header field indicates the initiator of the request. This may be different from the initiator of the dialog. Requests sent by the callee to the caller use the callee's address in the From header field.

The optional "display-name" is meant to be rendered by a human user interface. A system SHOULD use the display name "Anonymous" if the identity of the client is to remain hidden. Even if the "display-name" is empty, the "name-addr" form MUST be used if the "addr-spec" contains a comma, question mark, or semicolon. Syntax issues are discussed in Section 7.3.1.

Two From header fields are equivalent if their URIs match, and their parameters match. Extension parameters in one header field, not present in the other are ignored for the purposes of comparison. This means that the display name and presence or absence of angle brackets do not affect matching.

See Section 20.10 for the rules for parsing a display name, URI and URI parameters, and header field parameters.

The compact form of the From header field is f.

Examples:

```
From: "A. G. Bell" <sip:agb@bell-telephone.com> ;tag=a48s
From: sip:+12125551212@server.phone2net.com;tag=887s
f: Anonymous <sip:c8oqz84zk7z@privacy.org>;tag=hyh8
```

20.21 In-Reply-To

The In-Reply-To header field enumerates the Call-IDs that this call references or returns. These Call-IDs may have been cached by the client then included in this header field in a return call.

This allows automatic call distribution systems to route return calls to the originator of the first call. This also allows callees to filter calls, so that only return calls for calls they originated will be accepted. This field is not a substitute for request authentication.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

Example:

In-Reply-To: 70710@saturn.bell-tel.com, 17320@saturn.bell-tel.com

20.22 Max-Forwards

The Max-Forwards header field must be used with any SIP method to limit the number of proxies or gateways that can forward the request to the next downstream server. This can also be useful when the client is attempting to trace a request chain that appears to be failing or looping in mid-chain.

The Max-Forwards value is an integer in the range 0-255 indicating the remaining number of times this request message is allowed to be forwarded. This count is decremented by each server that forwards the request. The recommended initial value is 70.

This header field should be inserted by elements that can not otherwise guarantee loop detection. For example, a B2BUA should insert a Max-Forwards header field.

Example:

Max-Forwards: 6

20.23 Min-Expires

The Min-Expires header field conveys the minimum refresh interval supported for soft-state elements managed by that server. This includes Contact header fields that are stored by a registrar. The header field contains a decimal integer number of seconds from 0 to (2**32)-1. The use of the header field in a 423 (Interval Too Brief) response is described in Sections 10.2.8, 10.3, and 21.4.17.

Example:

Min-Expires: 60

20.24 MIME-Version

See [H19.4.1].

Example:

MIME-Version: 1.0



20.25 Organization

The Organization header field conveys the name of the organization to which the SIP element issuing the request or response belongs.

The field MAY be used by client software to filter calls.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 794
Keysight, EX-1045
Page 794

Example:

Organization: Boxes by Bob

20.26 Priority

The Priority header field indicates the urgency of the request as perceived by the client. The Priority header field describes the priority that the SIP request should have to the receiving human or its agent. For example, it may be factored into decisions about call routing and acceptance. For these decisions, a message containing no Priority header field SHOULD be treated as if it specified a Priority of "normal". The Priority header field does not influence the use of communications resources such as packet forwarding priority in routers or access to circuits in PSTN gateways. The header field can have the values "non-urgent", "normal", "urgent", and "emergency", but additional values can be defined elsewhere. It is RECOMMENDED that the value of "emergency" only be used when life, limb, or property are in imminent danger. Otherwise, there are no semantics defined for this header field.

These are the values of RFC 2076 [38], with the addition of "emergency".

Examples:

Subject: A tornado is heading our way!
Priority: emergency

or

Subject: Weekend plans
Priority: non-urgent

20.27 Proxy-Authenticate

A Proxy-Authenticate header field value contains an authentication challenge.

The use of this header field is defined in [H14.33]. See Section 22.3 for further details on its usage.





Example:

Proxy-Authenticate: Digest realm="atlanta.com",
domain="sip:ss1.carrier.com", qop="auth",
nonce="f84f1cec41e6cbe5aea9c8e88d359",
opaque="", stale=FALSE, algorithm=MD5

20.28 Proxy-Authorization

The Proxy-Authorization header field allows the client to identify itself (or its user) to a proxy that requires authentication. A Proxy-Authorization field value consists of credentials containing the authentication information of the user agent for the proxy and/or

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 795
Keysight, EX-1045
Page 795

realm of the resource being requested.

See Section 22.3 for a definition of the usage of this header field.

This header field, along with Authorization, breaks the general rules about multiple header field names. Although not a comma-separated list, this header field name may be present multiple times, and MUST NOT be combined into a single header line using the usual rules described in Section 7.3.1.

Example:

```
Proxy-Authorization: Digest username="Alice", realm="atlanta.com",
   nonce="c60f3082ee1212b402a21831ae",
   response="245f23415f11432b3434341c022"
```

### 20.29 Proxy-Require

The Proxy-Require header field is used to indicate proxy-sensitive features that must be supported by the proxy. See Section 20.32 for more details on the mechanics of this message and a usage example.

Example:

```
   Proxy-Require: foo
```

### 20.30 Record-Route

The Record-Route header field is inserted by proxies in a request to force future requests in the dialog to be routed through the proxy.

Examples of its use with the Route header field are described in Sections 16.12.1.





Example:

```
   Record-Route: <sip:server10.biloxi.com;lr>,
                 <sip:bigbox3.site3.atlanta.com;lr>
```

### 20.31 Reply-To

The Reply-To header field contains a logical return URI that may be different from the From header field. For example, the URI MAY be used to return missed calls or unestablished sessions. If the user wished to remain anonymous, the header field SHOULD either be omitted from the request or populated in such a way that does not reveal any private information.

Even if the "display-name" is empty, the "name-addr" form MUST be used if the "addr-spec" contains a comma, question mark, or semicolon. Syntax issues are discussed in Section 7.3.1.

Example:

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

```
      Reply-To: Bob <sip:bob@biloxi.com>
```

20.32 Require

The Require header field is used by UACs to tell UASs about options that the UAC expects the UAS to support in order to process the request. Although an optional header field, the Require MUST NOT be ignored if it is present.

The Require header field contains a list of option tags, described in Section 19.2. Each option tag defines a SIP extension that MUST be understood to process the request. Frequently, this is used to indicate that a specific set of extension header fields need to be understood. A UAC compliant to this specification MUST only include option tags corresponding to standards-track RFCs.

Example:

```
      Require: 100rel
```

20.33 Retry-After

The Retry-After header field can be used with a 500 (Server Internal Error) or 503 (Service Unavailable) response to indicate how long the service is expected to be unavailable to the requesting client and with a 404 (Not Found), 413 (Request Entity Too Large), 480 (Temporarily Unavailable), 486 (Busy Here), 600 (Busy), or 603





(Decline) response to indicate when the called party anticipates being available again. The value of this field is a positive integer number of seconds (in decimal) after the time of the response.

An optional comment can be used to indicate additional information about the time of callback. An optional "duration" parameter indicates how long the called party will be reachable starting at the initial time of availability. If no duration parameter is given, the service is assumed to be available indefinitely.

Examples:

```
      Retry-After: 18000;duration=3600
      Retry-After: 120 (I'm in a meeting)
```

20.34 Route

The Route header field is used to force routing for a request through the listed set of proxies. Examples of the use of the Route header field are in Section 16.12.1.

Example:

```
      Route: <sip:bigbox3.site3.atlanta.com;lr>,
             <sip:server10.biloxi.com;lr>
```

20.35 Server

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

The Server header field contains information about the software used by the UAS to handle the request.

Revealing the specific software version of the server might allow the server to become more vulnerable to attacks against software that is known to contain security holes. Implementers SHOULD make the Server header field a configurable option.

Example:

```
Server: HomeServer v2
```

### 20.36 Subject

The Subject header field provides a summary or indicates the nature of the call, allowing call filtering without having to parse the session description. The session description does not have to use the same subject indication as the invitation.

The compact form of the Subject header field is s.





Example:

```
Subject: Need more boxes
s: Tech Support
```

### 20.37 Supported

The Supported header field enumerates all the extensions supported by the UAC or UAS.

The Supported header field contains a list of option tags, described in Section 19.2, that are understood by the UAC or UAS. A UA compliant to this specification MUST only include option tags corresponding to standards-track RFCs. If empty, it means that no extensions are supported.

The compact form of the Supported header field is k.

Example:

```
Supported: 100rel
```

### 20.38 Timestamp

The Timestamp header field describes when the UAC sent the request to the UAS.

See Section 8.2.6 for details on how to generate a response to a request that contains the header field. Although there is no normative behavior defined here that makes use of the header, it allows for extensions or SIP applications to obtain RTT estimates.

Example:

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

```
      Timestamp: 54

20.39 To

   The To header field specifies the logical recipient of the request.

   The optional "display-name" is meant to be rendered by a human-user
   interface.  The "tag" parameter serves as a general mechanism for
   dialog identification.

   See Section 19.3 for details of the "tag" parameter.
```





```
   Comparison of To header fields for equality is identical to
   comparison of From header fields.  See Section 20.10 for the rules
   for parsing a display name, URI and URI parameters, and header field
   parameters.

   The compact form of the To header field is t.

   The following are examples of valid To header fields:

      To: The Operator <sip:operator@cs.columbia.edu>;tag=287447
      t: sip:+12125551212@server.phone2net.com

20.40 Unsupported

   The Unsupported header field lists the features not supported by the
   UAS.  See Section 20.32 for motivation.

   Example:

      Unsupported: foo

20.41 User-Agent

   The User-Agent header field contains information about the UAC
   originating the request.  The semantics of this header field are
   defined in [H14.43].

   Revealing the specific software version of the user agent might allow
   the user agent to become more vulnerable to attacks against software
   that is known to contain security holes.  Implementers SHOULD make
   the User-Agent header field a configurable option.

   Example:

      User-Agent: Softphone Beta1.5

20.42 Via

   The Via header field indicates the path taken by the request so far
   and indicates the path that should be followed in routing responses.
   The branch ID parameter in the Via header field values serves as a
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 799
Keysight, EX-1045
Page 799

transaction identifier, and is used by proxies to detect loops.

A Via header field value contains the transport protocol used to send the message, the client's host name or network address, and possibly the port number at which it wishes to receive responses. A Via header field value can also contain parameters such as "maddr", "ttl", "received", and "branch", whose meaning and use are described



in other sections. For implementations compliant to this specification, the value of the branch parameter MUST start with the magic cookie "z9hG4bK", as discussed in Section 8.1.1.7.

Transport protocols defined here are "UDP", "TCP", "TLS", and "SCTP". "TLS" means TLS over TCP. When a request is sent to a SIPS URI, the protocol still indicates "SIP", and the transport protocol is TLS.

```
Via: SIP/2.0/UDP erlang.bell-telephone.com:5060;branch=z9hG4bK87asdks7
Via: SIP/2.0/UDP 192.0.2.1:5060 ;received=192.0.2.207
     ;branch=z9hG4bK77asjd
```

The compact form of the Via header field is v.

In this example, the message originated from a multi-homed host with two addresses, 192.0.2.1 and 192.0.2.207. The sender guessed wrong as to which network interface would be used. Erlang.bell-telephone.com noticed the mismatch and added a parameter to the previous hop's Via header field value, containing the address that the packet actually came from.

The host or network address and port number are not required to follow the SIP URI syntax. Specifically, LWS on either side of the ":" or "/" is allowed, as shown here:

```
   Via: SIP / 2.0 / UDP first.example.com: 4000;ttl=16
     ;maddr=224.2.0.1 ;branch=z9hG4bKa7c6a8dlze.1
```

Even though this specification mandates that the branch parameter be present in all requests, the BNF for the header field indicates that it is optional. This allows interoperation with RFC 2543 elements, which did not have to insert the branch parameter.

Two Via header fields are equal if their sent-protocol and sent-by fields are equal, both have the same set of parameters, and the values of all parameters are equal.

### 20.43 Warning

The Warning header field is used to carry additional information about the status of a response. Warning header field values are sent with responses and contain a three-digit warning code, host name, and warning text.

The "warn-text" should be in a natural language that is most likely to be intelligible to the human user receiving the response. This decision can be based on any available knowledge, such as the location of the user, the Accept-Language field in a request, or the

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

Content-Language field in a response. The default language is i-default [21].

The currently-defined "warn-code"s are listed below, with a recommended warn-text in English and a description of their meaning. These warnings describe failures induced by the session description. The first digit of warning codes beginning with "3" indicates warnings specific to SIP. Warnings 300 through 329 are reserved for indicating problems with keywords in the session description, 330 through 339 are warnings related to basic network services requested in the session description, 370 through 379 are warnings related to quantitative QoS parameters requested in the session description, and 390 through 399 are miscellaneous warnings that do not fall into one of the above categories.

300 Incompatible network protocol: One or more network protocols contained in the session description are not available.

301 Incompatible network address formats: One or more network address formats contained in the session description are not available.

302 Incompatible transport protocol: One or more transport protocols described in the session description are not available.

303 Incompatible bandwidth units: One or more bandwidth measurement units contained in the session description were not understood.

304 Media type not available: One or more media types contained in the session description are not available.

305 Incompatible media format: One or more media formats contained in the session description are not available.

306 Attribute not understood: One or more of the media attributes in the session description are not supported.

307 Session description parameter not understood: A parameter other than those listed above was not understood.

330 Multicast not available: The site where the user is located does not support multicast.

331 Unicast not available: The site where the user is located does not support unicast communication (usually due to the presence of a firewall).



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 801
Keysight, EX-1045
Page 801

370 Insufficient bandwidth: The bandwidth specified in the session description or defined by the media exceeds that known to be available.

399 Miscellaneous warning: The warning text can include arbitrary information to be presented to a human user or logged. A system receiving this warning MUST NOT take any automated action.

1xx and 2xx have been taken by HTTP/1.1.

Additional "warn-code"s can be defined through IANA, as defined in Section 27.2.

Examples:

```
Warning: 307 isi.edu "Session parameter 'foo' not understood"
Warning: 301 isi.edu "Incompatible network address type 'E.164'"
```

### 20.44 WWW-Authenticate

A WWW-Authenticate header field value contains an authentication challenge. See Section 22.2 for further details on its usage.

Example:

```
WWW-Authenticate: Digest realm="atlanta.com",
  domain="sip:boxesbybob.com", qop="auth",
  nonce="f84f1cec41e6cbe5aea9c8e88d359",
  opaque="", stale=FALSE, algorithm=MD5
```

## 21 Response Codes

The response codes are consistent with, and extend, HTTP/1.1 response codes. Not all HTTP/1.1 response codes are appropriate, and only those that are appropriate are given here. Other HTTP/1.1 response codes SHOULD NOT be used. Also, SIP defines a new class, 6xx.

### 21.1 Provisional 1xx

Provisional responses, also known as informational responses, indicate that the server contacted is performing some further action and does not yet have a definitive response. A server sends a 1xx response if it expects to take more than 200 ms to obtain a final response. Note that 1xx responses are not transmitted reliably. They never cause the client to send an ACK. Provisional (1xx) responses MAY contain message bodies, including session descriptions.



#### 21.1.1 100 Trying

This response indicates that the request has been received by the next-hop server and that some unspecified action is being taken on behalf of this call (for example, a database is being consulted). This response, like all other provisional responses, stops

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 802
Keysight, EX-1045
Page 802

retransmissions of an INVITE by a UAC. The 100 (Trying) response is different from other provisional responses, in that it is never forwarded upstream by a stateful proxy.

21.1.2 180 Ringing

The UA receiving the INVITE is trying to alert the user. This response MAY be used to initiate local ringback.

21.1.3 181 Call Is Being Forwarded

A server MAY use this status code to indicate that the call is being forwarded to a different set of destinations.

21.1.4 182 Queued

The called party is temporarily unavailable, but the server has decided to queue the call rather than reject it. When the callee becomes available, it will return the appropriate final status response. The reason phrase MAY give further details about the status of the call, for example, "5 calls queued; expected waiting time is 15 minutes". The server MAY issue several 182 (Queued) responses to update the caller about the status of the queued call.

21.1.5 183 Session Progress

The 183 (Session Progress) response is used to convey information about the progress of the call that is not otherwise classified. The Reason-Phrase, header fields, or message body MAY be used to convey more details about the call progress.

21.2 Successful 2xx

The request was successful.

21.2.1 200 OK

The request has succeeded. The information returned with the response depends on the method used in the request.





21.3 Redirection 3xx

3xx responses give information about the user's new location, or about alternative services that might be able to satisfy the call.

21.3.1 300 Multiple Choices

The address in the request resolved to several choices, each with its own specific location, and the user (or UA) can select a preferred communication end point and redirect its request to that location.

The response MAY include a message body containing a list of resource characteristics and location(s) from which the user or UA can choose

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 803
Keysight, EX-1045
Page 803

the one most appropriate, if allowed by the Accept request header field. However, no MIME types have been defined for this message body.

The choices SHOULD also be listed as Contact fields (Section 20.10). Unlike HTTP, the SIP response MAY contain several Contact fields or a list of addresses in a Contact field. UAs MAY use the Contact header field value for automatic redirection or MAY ask the user to confirm a choice. However, this specification does not define any standard for such automatic selection.

This status response is appropriate if the callee can be reached at several different locations and the server cannot or prefers not to proxy the request.

21.3.2 301 Moved Permanently

The user can no longer be found at the address in the Request-URI, and the requesting client SHOULD retry at the new address given by the Contact header field (Section 20.10). The requestor SHOULD update any local directories, address books, and user location caches with this new value and redirect future requests to the address(es) listed.

21.3.3 302 Moved Temporarily

The requesting client SHOULD retry the request at the new address(es) given by the Contact header field (Section 20.10). The Request-URI of the new request uses the value of the Contact header field in the response.





The duration of the validity of the Contact URI can be indicated through an Expires (Section 20.19) header field or an expires parameter in the Contact header field. Both proxies and UAs MAY cache this URI for the duration of the expiration time. If there is no explicit expiration time, the address is only valid once for recursing, and MUST NOT be cached for future transactions.

If the URI cached from the Contact header field fails, the Request-URI from the redirected request MAY be tried again a single time.

The temporary URI may have become out-of-date sooner than the expiration time, and a new temporary URI may be available.

21.3.4 305 Use Proxy

The requested resource MUST be accessed through the proxy given by the Contact field. The Contact field gives the URI of the proxy. The recipient is expected to repeat this single request via the proxy. 305 (Use Proxy) responses MUST only be generated by UASs.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

21.3.5 380 Alternative Service

The call was not successful, but alternative services are possible.

The alternative services are described in the message body of the response. Formats for such bodies are not defined here, and may be the subject of future standardization.

21.4 Request Failure 4xx

4xx responses are definite failure responses from a particular server. The client SHOULD NOT retry the same request without modification (for example, adding appropriate authorization). However, the same request to a different server might be successful.

21.4.1 400 Bad Request

The request could not be understood due to malformed syntax. The Reason-Phrase SHOULD identify the syntax problem in more detail, for example, "Missing Call-ID header field".

21.4.2 401 Unauthorized

The request requires user authentication. This response is issued by UASs and registrars, while 407 (Proxy Authentication Required) is used by proxy servers.





21.4.3 402 Payment Required

Reserved for future use.

21.4.4 403 Forbidden

The server understood the request, but is refusing to fulfill it. Authorization will not help, and the request SHOULD NOT be repeated.

21.4.5 404 Not Found

The server has definitive information that the user does not exist at the domain specified in the Request-URI. This status is also returned if the domain in the Request-URI does not match any of the domains handled by the recipient of the request.

21.4.6 405 Method Not Allowed

The method specified in the Request-Line is understood, but not allowed for the address identified by the Request-URI.

The response MUST include an Allow header field containing a list of valid methods for the indicated address.

21.4.7 406 Not Acceptable

The resource identified by the request is only capable of generating

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

response entities that have content characteristics not acceptable according to the Accept header field sent in the request.

21.4.8 407 Proxy Authentication Required

This code is similar to 401 (Unauthorized), but indicates that the client MUST first authenticate itself with the proxy. SIP access authentication is explained in Sections 26 and 22.3.

This status code can be used for applications where access to the communication channel (for example, a telephony gateway) rather than the callee requires authentication.

21.4.9 408 Request Timeout

The server could not produce a response within a suitable amount of time, for example, if it could not determine the location of the user in time. The client MAY repeat the request without modifications at any later time.





21.4.10 410 Gone

The requested resource is no longer available at the server and no forwarding address is known. This condition is expected to be considered permanent. If the server does not know, or has no facility to determine, whether or not the condition is permanent, the status code 404 (Not Found) SHOULD be used instead.

21.4.11 413 Request Entity Too Large

The server is refusing to process a request because the request entity-body is larger than the server is willing or able to process. The server MAY close the connection to prevent the client from continuing the request.

If the condition is temporary, the server SHOULD include a Retry-After header field to indicate that it is temporary and after what time the client MAY try again.

21.4.12 414 Request-URI Too Long

The server is refusing to service the request because the Request-URI is longer than the server is willing to interpret.

21.4.13 415 Unsupported Media Type

The server is refusing to service the request because the message body of the request is in a format not supported by the server for the requested method. The server MUST return a list of acceptable formats using the Accept, Accept-Encoding, or Accept-Language header field, depending on the specific problem with the content. UAC processing of this response is described in Section 8.1.3.5.

21.4.14 416 Unsupported URI Scheme

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 806
Keysight, EX-1045
Page 806

The server cannot process the request because the scheme of the URI in the Request-URI is unknown to the server. Client processing of this response is described in Section 8.1.3.5.

21.4.15 420 Bad Extension

The server did not understand the protocol extension specified in a Proxy-Require (Section 20.29) or Require (Section 20.32) header field. The server MUST include a list of the unsupported extensions in an Unsupported header field in the response. UAC processing of this response is described in Section 8.1.3.5.





21.4.16 421 Extension Required

The UAS needs a particular extension to process the request, but this extension is not listed in a Supported header field in the request. Responses with this status code MUST contain a Require header field listing the required extensions.

A UAS SHOULD NOT use this response unless it truly cannot provide any useful service to the client. Instead, if a desirable extension is not listed in the Supported header field, servers SHOULD process the request using baseline SIP capabilities and any extensions supported by the client.

21.4.17 423 Interval Too Brief

The server is rejecting the request because the expiration time of the resource refreshed by the request is too short. This response can be used by a registrar to reject a registration whose Contact header field expiration time was too small. The use of this response and the related Min-Expires header field are described in Sections 10.2.8, 10.3, and 20.23.

21.4.18 480 Temporarily Unavailable

The callee's end system was contacted successfully but the callee is currently unavailable (for example, is not logged in, logged in but in a state that precludes communication with the callee, or has activated the "do not disturb" feature). The response MAY indicate a better time to call in the Retry-After header field. The user could also be available elsewhere (unbeknownst to this server). The reason phrase SHOULD indicate a more precise cause as to why the callee is unavailable. This value SHOULD be settable by the UA. Status 486 (Busy Here) MAY be used to more precisely indicate a particular reason for the call failure.

This status is also returned by a redirect or proxy server that recognizes the user identified by the Request-URI, but does not currently have a valid forwarding location for that user.

21.4.19 481 Call/Transaction Does Not Exist

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 807
Keysight, EX-1045
Page 807

This status indicates that the UAS received a request that does not match any existing dialog or transaction.

21.4.20 482 Loop Detected

The server has detected a loop (Section 16.3 Item 4).





21.4.21 483 Too Many Hops

The server received a request that contains a Max-Forwards (Section 20.22) header field with the value zero.

21.4.22 484 Address Incomplete

The server received a request with a Request-URI that was incomplete. Additional information SHOULD be provided in the reason phrase.

This status code allows overlapped dialing. With overlapped dialing, the client does not know the length of the dialing string. It sends strings of increasing lengths, prompting the user for more input, until it no longer receives a 484 (Address Incomplete) status response.

21.4.23 485 Ambiguous

The Request-URI was ambiguous. The response MAY contain a listing of possible unambiguous addresses in Contact header fields. Revealing alternatives can infringe on privacy of the user or the organization. It MUST be possible to configure a server to respond with status 404 (Not Found) or to suppress the listing of possible choices for ambiguous Request-URIs.

Example response to a request with the Request-URI sip:lee@example.com:

```
SIP/2.0 485 Ambiguous
Contact: Carol Lee <sip:carol.lee@example.com>
Contact: Ping Lee <sip:p.lee@example.com>
Contact: Lee M. Foote <sips:lee.foote@example.com>
```

Some email and voice mail systems provide this functionality. A status code separate from 3xx is used since the semantics are different: for 300, it is assumed that the same person or service will be reached by the choices provided. While an automated choice or sequential search makes sense for a 3xx response, user intervention is required for a 485 (Ambiguous) response.

21.4.24 486 Busy Here

The callee's end system was contacted successfully, but the callee is currently not willing or able to take additional calls at this end system. The response MAY indicate a better time to call in the Retry-After header field. The user could also be available

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 808
Keysight, EX-1045
Page 808

elsewhere, such as through a voice mail service. Status 600 (Busy Everywhere) SHOULD be used if the client knows that no other end system will be able to accept this call.

21.4.25 487 Request Terminated

The request was terminated by a BYE or CANCEL request. This response is never returned for a CANCEL request itself.

21.4.26 488 Not Acceptable Here

The response has the same meaning as 606 (Not Acceptable), but only applies to the specific resource addressed by the Request-URI and the request may succeed elsewhere.

A message body containing a description of media capabilities MAY be present in the response, which is formatted according to the Accept header field in the INVITE (or application/sdp if not present), the same as a message body in a 200 (OK) response to an OPTIONS request.

21.4.27 491 Request Pending

The request was received by a UAS that had a pending request within the same dialog. Section 14.2 describes how such "glare" situations are resolved.

21.4.28 493 Undecipherable

The request was received by a UAS that contained an encrypted MIME body for which the recipient does not possess or will not provide an appropriate decryption key. This response MAY have a single body containing an appropriate public key that should be used to encrypt MIME bodies sent to this UA. Details of the usage of this response code can be found in Section 23.2.

21.5 Server Failure 5xx

5xx responses are failure responses given when a server itself has erred.

21.5.1 500 Server Internal Error

The server encountered an unexpected condition that prevented it from fulfilling the request. The client MAY display the specific error condition and MAY retry the request after several seconds.

If the condition is temporary, the server MAY indicate when the client may retry the request using the Retry-After header field.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 809
Keysight, EX-1045
Page 809

21.5.2 501 Not Implemented

The server does not support the functionality required to fulfill the request. This is the appropriate response when a UAS does not recognize the request method and is not capable of supporting it for any user. (Proxies forward all requests regardless of method.)

Note that a 405 (Method Not Allowed) is sent when the server recognizes the request method, but that method is not allowed or supported.

21.5.3 502 Bad Gateway

The server, while acting as a gateway or proxy, received an invalid response from the downstream server it accessed in attempting to fulfill the request.

21.5.4 503 Service Unavailable

The server is temporarily unable to process the request due to a temporary overloading or maintenance of the server. The server MAY indicate when the client should retry the request in a Retry-After header field. If no Retry-After is given, the client MUST act as if it had received a 500 (Server Internal Error) response.

A client (proxy or UAC) receiving a 503 (Service Unavailable) SHOULD attempt to forward the request to an alternate server. It SHOULD NOT forward any other requests to that server for the duration specified in the Retry-After header field, if present.

Servers MAY refuse the connection or drop the request instead of responding with 503 (Service Unavailable).

21.5.5 504 Server Time-out

The server did not receive a timely response from an external server it accessed in attempting to process the request. 408 (Request Timeout) should be used instead if there was no response within the period specified in the Expires header field from the upstream server.

21.5.6 505 Version Not Supported

The server does not support, or refuses to support, the SIP protocol version that was used in the request. The server is indicating that it is unable or unwilling to complete the request using the same major version as the client, other than with this error message.



21.5.7 513 Message Too Large

The server was unable to process the request since the message length exceeded its capabilities.

21.6 Global Failures 6xx

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 810
Keysight, EX-1045
Page 810

6xx responses indicate that a server has definitive information about a particular user, not just the particular instance indicated in the Request-URI.

21.6.1 600 Busy Everywhere

The callee's end system was contacted successfully but the callee is busy and does not wish to take the call at this time. The response MAY indicate a better time to call in the Retry-After header field. If the callee does not wish to reveal the reason for declining the call, the callee uses status code 603 (Decline) instead. This status response is returned only if the client knows that no other end point (such as a voice mail system) will answer the request. Otherwise, 486 (Busy Here) should be returned.

21.6.2 603 Decline

The callee's machine was successfully contacted but the user explicitly does not wish to or cannot participate. The response MAY indicate a better time to call in the Retry-After header field. This status response is returned only if the client knows that no other end point will answer the request.

21.6.3 604 Does Not Exist Anywhere

The server has authoritative information that the user indicated in the Request-URI does not exist anywhere.

21.6.4 606 Not Acceptable

The user's agent was contacted successfully but some aspects of the session description such as the requested media, bandwidth, or addressing style were not acceptable.

A 606 (Not Acceptable) response means that the user wishes to communicate, but cannot adequately support the session described. The 606 (Not Acceptable) response MAY contain a list of reasons in a Warning header field describing why the session described cannot be supported. Warning reason codes are listed in Section 20.43.





A message body containing a description of media capabilities MAY be present in the response, which is formatted according to the Accept header field in the INVITE (or application/sdp if not present), the same as a message body in a 200 (OK) response to an OPTIONS request.

It is hoped that negotiation will not frequently be needed, and when a new user is being invited to join an already existing conference, negotiation may not be possible. It is up to the invitation initiator to decide whether or not to act on a 606 (Not Acceptable) response.

This status response is returned only if the client knows that no other end point will answer the request.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

## 22 Usage of HTTP Authentication

SIP provides a stateless, challenge-based mechanism for authentication that is based on authentication in HTTP. Any time that a proxy server or UA receives a request (with the exceptions given in Section 22.1), it MAY challenge the initiator of the request to provide assurance of its identity. Once the originator has been identified, the recipient of the request SHOULD ascertain whether or not this user is authorized to make the request in question. No authorization systems are recommended or discussed in this document.

The "Digest" authentication mechanism described in this section provides message authentication and replay protection only, without message integrity or confidentiality. Protective measures above and beyond those provided by Digest need to be taken to prevent active attackers from modifying SIP requests and responses.

Note that due to its weak security, the usage of "Basic" authentication has been deprecated. Servers MUST NOT accept credentials using the "Basic" authorization scheme, and servers also MUST NOT challenge with "Basic". This is a change from RFC 2543.

### 22.1 Framework

The framework for SIP authentication closely parallels that of HTTP (RFC 2617 [17]). In particular, the BNF for auth-scheme, auth-param, challenge, realm, realm-value, and credentials is identical (although the usage of "Basic" as a scheme is not permitted). In SIP, a UAS uses the 401 (Unauthorized) response to challenge the identity of a UAC. Additionally, registrars and redirect servers MAY make use of 401 (Unauthorized) responses for authentication, but proxies MUST NOT, and instead MAY use the 407 (Proxy Authentication Required)





response. The requirements for inclusion of the Proxy-Authenticate, Proxy-Authorization, WWW-Authenticate, and Authorization in the various messages are identical to those described in RFC 2617 [17].

Since SIP does not have the concept of a canonical root URL, the notion of protection spaces is interpreted differently in SIP. The realm string alone defines the protection domain. This is a change from RFC 2543, in which the Request-URI and the realm together defined the protection domain.

> This previous definition of protection domain caused some amount of confusion since the Request-URI sent by the UAC and the Request-URI received by the challenging server might be different, and indeed the final form of the Request-URI might not be known to the UAC. Also, the previous definition depended on the presence of a SIP URI in the Request-URI and seemed to rule out alternative URI schemes (for example, the tel URL).

Operators of user agents or proxy servers that will authenticate received requests MUST adhere to the following guidelines for

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 812
Keysight, EX-1045
Page 812

creation of a realm string for their server:

- Realm strings MUST be globally unique. It is RECOMMENDED that a realm string contain a hostname or domain name, following the recommendation in Section 3.2.1 of RFC 2617 [17].

- Realm strings SHOULD present a human-readable identifier that can be rendered to a user.

For example:

```
INVITE sip:bob@biloxi.com SIP/2.0
Authorization: Digest realm="biloxi.com", <...>
```

Generally, SIP authentication is meaningful for a specific realm, a protection domain. Thus, for Digest authentication, each such protection domain has its own set of usernames and passwords. If a server does not require authentication for a particular request, it MAY accept a default username, "anonymous", which has no password (password of ""). Similarly, UACs representing many users, such as PSTN gateways, MAY have their own device-specific username and password, rather than accounts for particular users, for their realm.

While a server can legitimately challenge most SIP requests, there are two requests defined by this document that require special handling for authentication: ACK and CANCEL.





Under an authentication scheme that uses responses to carry values used to compute nonces (such as Digest), some problems come up for any requests that take no response, including ACK. For this reason, any credentials in the INVITE that were accepted by a server MUST be accepted by that server for the ACK. UACs creating an ACK message will duplicate all of the Authorization and Proxy-Authorization header field values that appeared in the INVITE to which the ACK corresponds. Servers MUST NOT attempt to challenge an ACK.

Although the CANCEL method does take a response (a 2xx), servers MUST NOT attempt to challenge CANCEL requests since these requests cannot be resubmitted. Generally, a CANCEL request SHOULD be accepted by a server if it comes from the same hop that sent the request being canceled (provided that some sort of transport or network layer security association, as described in Section 26.2.1, is in place).

When a UAC receives a challenge, it SHOULD render to the user the contents of the "realm" parameter in the challenge (which appears in either a WWW-Authenticate header field or Proxy-Authenticate header field) if the UAC device does not already know of a credential for the realm in question. A service provider that pre-configures UAs with credentials for its realm should be aware that users will not have the opportunity to present their own credentials for this realm when challenged at a pre-configured device.

Finally, note that even if a UAC can locate credentials that are associated with the proper realm, the potential exists that these

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 813
Keysight, EX-1045
Page 813

credentials may no longer be valid or that the challenging server will not accept these credentials for whatever reason (especially when "anonymous" with no password is submitted). In this instance a server may repeat its challenge, or it may respond with a 403 Forbidden. A UAC MUST NOT re-attempt requests with the credentials that have just been rejected (though the request may be retried if the nonce was stale).

### 22.2 User-to-User Authentication

When a UAS receives a request from a UAC, the UAS MAY authenticate the originator before the request is processed. If no credentials (in the Authorization header field) are provided in the request, the UAS can challenge the originator to provide credentials by rejecting the request with a 401 (Unauthorized) status code.

The WWW-Authenticate response-header field MUST be included in 401 (Unauthorized) response messages. The field value consists of at least one challenge that indicates the authentication scheme(s) and parameters applicable to the realm.





An example of the WWW-Authenticate header field in a 401 challenge is:

```
WWW-Authenticate: Digest
        realm="biloxi.com",
        qop="auth,auth-int",
        nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
        opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

When the originating UAC receives the 401 (Unauthorized), it SHOULD, if it is able, re-originate the request with the proper credentials. The UAC may require input from the originating user before proceeding. Once authentication credentials have been supplied (either directly by the user, or discovered in an internal keyring), UAs SHOULD cache the credentials for a given value of the To header field and "realm" and attempt to re-use these values on the next request for that destination. UAs MAY cache credentials in any way they would like.

If no credentials for a realm can be located, UACs MAY attempt to retry the request with a username of "anonymous" and no password (a password of "").

Once credentials have been located, any UA that wishes to authenticate itself with a UAS or registrar -- usually, but not necessarily, after receiving a 401 (Unauthorized) response -- MAY do so by including an Authorization header field with the request. The Authorization field value consists of credentials containing the authentication information of the UA for the realm of the resource being requested as well as parameters required in support of authentication and replay protection.

An example of the Authorization header field is:

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 814
Keysight, EX-1045
Page 814

```
Authorization: Digest username="bob",
        realm="biloxi.com",
        nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
        uri="sip:bob@biloxi.com",
        qop=auth,
        nc=00000001,
        cnonce="0a4f113b",
        response="6629fae49393a05397450978507c4ef1",
        opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

When a UAC resubmits a request with its credentials after receiving a 401 (Unauthorized) or 407 (Proxy Authentication Required) response, it MUST increment the CSeq header field value as it would normally when sending an updated request.





### 22.3 Proxy-to-User Authentication

Similarly, when a UAC sends a request to a proxy server, the proxy server MAY authenticate the originator before the request is processed. If no credentials (in the Proxy-Authorization header field) are provided in the request, the proxy can challenge the originator to provide credentials by rejecting the request with a 407 (Proxy Authentication Required) status code. The proxy MUST populate the 407 (Proxy Authentication Required) message with a Proxy-Authenticate header field value applicable to the proxy for the requested resource.

The use of Proxy-Authenticate and Proxy-Authorization parallel that described in [17], with one difference. Proxies MUST NOT add values to the Proxy-Authorization header field. All 407 (Proxy Authentication Required) responses MUST be forwarded upstream toward the UAC following the procedures for any other response. It is the UAC's responsibility to add the Proxy-Authorization header field value containing credentials for the realm of the proxy that has asked for authentication.

> If a proxy were to resubmit a request adding a Proxy-Authorization header field value, it would need to increment the CSeq in the new request. However, this would cause the UAC that submitted the original request to discard a response from the UAS, as the CSeq value would be different.

When the originating UAC receives the 407 (Proxy Authentication Required) it SHOULD, if it is able, re-originate the request with the proper credentials. It should follow the same procedures for the display of the "realm" parameter that are given above for responding to 401.

If no credentials for a realm can be located, UACs MAY attempt to retry the request with a username of "anonymous" and no password (a password of "").

The UAC SHOULD also cache the credentials used in the re-originated request.

The following rule is RECOMMENDED for proxy credential caching:

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 815
Keysight, EX-1045
Page 815

If a UA receives a Proxy-Authenticate header field value in a 401/407 response to a request with a particular Call-ID, it should incorporate credentials for that realm in all subsequent requests that contain the same Call-ID. These credentials MUST NOT be cached across dialogs; however, if a UA is configured with the realm of its local outbound proxy, when one exists, then the UA MAY cache



credentials for that realm across dialogs. Note that this does mean a future request in a dialog could contain credentials that are not needed by any proxy along the Route header path.

Any UA that wishes to authenticate itself to a proxy server -- usually, but not necessarily, after receiving a 407 (Proxy Authentication Required) response -- MAY do so by including a Proxy-Authorization header field value with the request. The Proxy-Authorization request-header field allows the client to identify itself (or its user) to a proxy that requires authentication. The Proxy-Authorization header field value consists of credentials containing the authentication information of the UA for the proxy and/or realm of the resource being requested.

A Proxy-Authorization header field value applies only to the proxy whose realm is identified in the "realm" parameter (this proxy may previously have demanded authentication using the Proxy-Authenticate field). When multiple proxies are used in a chain, a Proxy-Authorization header field value MUST NOT be consumed by any proxy whose realm does not match the "realm" parameter specified in that value.

Note that if an authentication scheme that does not support realms is used in the Proxy-Authorization header field, a proxy server MUST attempt to parse all Proxy-Authorization header field values to determine whether one of them has what the proxy server considers to be valid credentials. Because this is potentially very time-consuming in large networks, proxy servers SHOULD use an authentication scheme that supports realms in the Proxy-Authorization header field.

If a request is forked (as described in Section 16.7), various proxy servers and/or UAs may wish to challenge the UAC. In this case, the forking proxy server is responsible for aggregating these challenges into a single response. Each WWW-Authenticate and Proxy-Authenticate value received in responses to the forked request MUST be placed into the single response that is sent by the forking proxy to the UA; the ordering of these header field values is not significant.

When a proxy server issues a challenge in response to a request, it will not proxy the request until the UAC has retried the request with valid credentials. A forking proxy may forward a request simultaneously to multiple proxy servers that require authentication, each of which in turn will not forward the request until the originating UAC has authenticated itself in their respective realm. If the UAC does not provide credentials for

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

each challenge, the proxy servers that issued the challenges will not forward requests to the UA where the destination user might be located, and therefore, the virtues of forking are largely lost.

When resubmitting its request in response to a 401 (Unauthorized) or 407 (Proxy Authentication Required) that contains multiple challenges, a UAC MAY include an Authorization value for each WWW-Authenticate value and a Proxy-Authorization value for each Proxy-Authenticate value for which the UAC wishes to supply a credential. As noted above, multiple credentials in a request SHOULD be differentiated by the "realm" parameter.

It is possible for multiple challenges associated with the same realm to appear in the same 401 (Unauthorized) or 407 (Proxy Authentication Required). This can occur, for example, when multiple proxies within the same administrative domain, which use a common realm, are reached by a forking request. When it retries a request, a UAC MAY therefore supply multiple credentials in Authorization or Proxy-Authorization header fields with the same "realm" parameter value. The same credentials SHOULD be used for the same realm.

## 22.4 The Digest Authentication Scheme

This section describes the modifications and clarifications required to apply the HTTP Digest authentication scheme to SIP. The SIP scheme usage is almost completely identical to that for HTTP [17].

Since RFC 2543 is based on HTTP Digest as defined in RFC 2069 [39], SIP servers supporting RFC 2617 MUST ensure they are backwards compatible with RFC 2069. Procedures for this backwards compatibility are specified in RFC 2617. Note, however, that SIP servers MUST NOT accept or request Basic authentication.

The rules for Digest authentication follow those defined in [17], with "HTTP/1.1" replaced by "SIP/2.0" in addition to the following differences:

1. The URI included in the challenge has the following BNF:

   URI = SIP-URI / SIPS-URI

2. The BNF in RFC 2617 has an error in that the 'uri' parameter of the Authorization header field for HTTP Digest



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

authentication is not enclosed in quotation marks. (The example in Section 3.5 of RFC 2617 is correct.) For SIP, the 'uri' MUST be enclosed in quotation marks.

3. The BNF for digest-uri-value is:

   digest-uri-value = Request-URI ; as defined in Section 25

4. The example procedure for choosing a nonce based on Etag does not work for SIP.

5. The text in RFC 2617 [17] regarding cache operation does not apply to SIP.

6. RFC 2617 [17] requires that a server check that the URI in the request line and the URI included in the Authorization header field point to the same resource. In a SIP context, these two URIs may refer to different users, due to forwarding at some proxy. Therefore, in SIP, a server MAY check that the Request-URI in the Authorization header field value corresponds to a user for whom the server is willing to accept forwarded or direct requests, but it is not necessarily a failure if the two fields are not equivalent.

7. As a clarification to the calculation of the A2 value for message integrity assurance in the Digest authentication scheme, implementers should assume, when the entity-body is empty (that is, when SIP messages have no body) that the hash of the entity-body resolves to the MD5 hash of an empty string, or:

   H(entity-body) = MD5("") =
"d41d8cd98f00b204e9800998ecf8427e"

8. RFC 2617 notes that a cnonce value MUST NOT be sent in an Authorization (and by extension Proxy-Authorization) header field if no qop directive has been sent. Therefore, any algorithms that have a dependency on the cnonce (including "MD5-Sess") require that the qop directive be sent. Use of the "qop" parameter is optional in RFC 2617 for the purposes of backwards compatibility with RFC 2069; since RFC 2543 was based on RFC 2069, the "qop" parameter must unfortunately remain optional for clients and servers to receive. However, servers MUST always send a "qop" parameter in WWW-Authenticate and Proxy-Authenticate header field values. If a client receives a "qop" parameter in a challenge header field, it MUST send the "qop" parameter in any resulting authorization header field.



RFC 2543 did not allow usage of the Authentication-Info header field (it effectively used RFC 2069). However, we now allow usage of this header field, since it provides integrity checks over the bodies and provides mutual authentication. RFC 2617 [17] defines mechanisms for backwards compatibility using the qop attribute in the request. These mechanisms MUST be used by a server to determine if the client

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 818
Keysight, EX-1045
Page 818

supports the new mechanisms in RFC 2617 that were not specified in RFC 2069.

## 23 S/MIME

SIP messages carry MIME bodies and the MIME standard includes mechanisms for securing MIME contents to ensure both integrity and confidentiality (including the 'multipart/signed' and 'application/pkcs7-mime' MIME types, see RFC 1847 [22], RFC 2630 [23] and RFC 2633 [24]). Implementers should note, however, that there may be rare network intermediaries (not typical proxy servers) that rely on viewing or modifying the bodies of SIP messages (especially SDP), and that secure MIME may prevent these sorts of intermediaries from functioning.

This applies particularly to certain types of firewalls.

The PGP mechanism for encrypting the header fields and bodies of SIP messages described in RFC 2543 has been deprecated.

### 23.1 S/MIME Certificates

The certificates that are used to identify an end-user for the purposes of S/MIME differ from those used by servers in one important respect - rather than asserting that the identity of the holder corresponds to a particular hostname, these certificates assert that the holder is identified by an end-user address. This address is composed of the concatenation of the "userinfo" "@" and "domainname" portions of a SIP or SIPS URI (in other words, an email address of the form "bob@biloxi.com"), most commonly corresponding to a user's address-of-record.

These certificates are also associated with keys that are used to sign or encrypt bodies of SIP messages. Bodies are signed with the private key of the sender (who may include their public key with the message as appropriate), but bodies are encrypted with the public key of the intended recipient. Obviously, senders must have foreknowledge of the public key of recipients in order to encrypt message bodies. Public keys can be stored within a UA on a virtual keyring.





Each user agent that supports S/MIME MUST contain a keyring specifically for end-users' certificates. This keyring should map between addresses of record and corresponding certificates. Over time, users SHOULD use the same certificate when they populate the originating URI of signaling (the From header field) with the same address-of-record.

Any mechanisms depending on the existence of end-user certificates are seriously limited in that there is virtually no consolidated authority today that provides certificates for end-user applications. However, users SHOULD acquire certificates from known public certificate authorities. As an alternative, users MAY create self-signed certificates. The implications of self-signed certificates

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 819
Keysight, EX-1045
Page 819

are explored further in Section 26.4.2. Implementations may also use pre-configured certificates in deployments in which a previous trust relationship exists between all SIP entities.

Above and beyond the problem of acquiring an end-user certificate, there are few well-known centralized directories that distribute end-user certificates. However, the holder of a certificate SHOULD publish their certificate in any public directories as appropriate. Similarly, UACs SHOULD support a mechanism for importing (manually or automatically) certificates discovered in public directories corresponding to the target URIs of SIP requests.

### 23.2 S/MIME Key Exchange

SIP itself can also be used as a means to distribute public keys in the following manner.

Whenever the CMS SignedData message is used in S/MIME for SIP, it MUST contain the certificate bearing the public key necessary to verify the signature.

When a UAC sends a request containing an S/MIME body that initiates a dialog, or sends a non-INVITE request outside the context of a dialog, the UAC SHOULD structure the body as an S/MIME 'multipart/signed' CMS SignedData body. If the desired CMS service is EnvelopedData (and the public key of the target user is known), the UAC SHOULD send the EnvelopedData message encapsulated within a SignedData message.

When a UAS receives a request containing an S/MIME CMS body that includes a certificate, the UAS SHOULD first validate the certificate, if possible, with any available root certificates for certificate authorities. The UAS SHOULD also determine the subject of the certificate (for S/MIME, the SubjectAltName will contain the appropriate identity) and compare this value to the From header field



of the request. If the certificate cannot be verified, because it is self-signed, or signed by no known authority, or if it is verifiable but its subject does not correspond to the From header field of request, the UAS MUST notify its user of the status of the certificate (including the subject of the certificate, its signer, and any key fingerprint information) and request explicit permission before proceeding. If the certificate was successfully verified and the subject of the certificate corresponds to the From header field of the SIP request, or if the user (after notification) explicitly authorizes the use of the certificate, the UAS SHOULD add this certificate to a local keyring, indexed by the address-of-record of the holder of the certificate.

When a UAS sends a response containing an S/MIME body that answers the first request in a dialog, or a response to a non-INVITE request outside the context of a dialog, the UAS SHOULD structure the body as an S/MIME 'multipart/signed' CMS SignedData body. If the desired CMS service is EnvelopedData, the UAS SHOULD send the EnvelopedData message encapsulated within a SignedData message.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 820
Keysight, EX-1045
Page 820

When a UAC receives a response containing an S/MIME CMS body that includes a certificate, the UAC SHOULD first validate the certificate, if possible, with any appropriate root certificate. The UAC SHOULD also determine the subject of the certificate and compare this value to the To field of the response; although the two may very well be different, and this is not necessarily indicative of a security breach. If the certificate cannot be verified because it is self-signed, or signed by no known authority, the UAC MUST notify its user of the status of the certificate (including the subject of the certificate, its signator, and any key fingerprint information) and request explicit permission before proceeding. If the certificate was successfully verified, and the subject of the certificate corresponds to the To header field in the response, or if the user (after notification) explicitly authorizes the use of the certificate, the UAC SHOULD add this certificate to a local keyring, indexed by the address-of-record of the holder of the certificate. If the UAC had not transmitted its own certificate to the UAS in any previous transaction, it SHOULD use a CMS SignedData body for its next request or response.

On future occasions, when the UA receives requests or responses that contain a From header field corresponding to a value in its keyring, the UA SHOULD compare the certificate offered in these messages with the existing certificate in its keyring. If there is a discrepancy, the UA MUST notify its user of a change of the certificate (preferably in terms that indicate that this is a potential security breach) and acquire the user's permission before continuing to





process the signaling. If the user authorizes this certificate, it SHOULD be added to the keyring alongside any previous value(s) for this address-of-record.

Note well however, that this key exchange mechanism does not guarantee the secure exchange of keys when self-signed certificates, or certificates signed by an obscure authority, are used - it is vulnerable to well-known attacks. In the opinion of the authors, however, the security it provides is proverbially better than nothing; it is in fact comparable to the widely used SSH application. These limitations are explored in greater detail in Section 26.4.2.

If a UA receives an S/MIME body that has been encrypted with a public key unknown to the recipient, it MUST reject the request with a 493 (Undecipherable) response. This response SHOULD contain a valid certificate for the respondent (corresponding, if possible, to any address of record given in the To header field of the rejected request) within a MIME body with a 'certs-only' "smime-type" parameter.

A 493 (Undecipherable) sent without any certificate indicates that the respondent cannot or will not utilize S/MIME encrypted messages, though they may still support S/MIME signatures.

Note that a user agent that receives a request containing an S/MIME body that is not optional (with a Content-Disposition header "handling" parameter of "required") MUST reject the request with a

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 821
Keysight, EX-1045
Page 821

415 Unsupported Media Type response if the MIME type is not understood. A user agent that receives such a response when S/MIME is sent SHOULD notify its user that the remote device does not support S/MIME, and it MAY subsequently resend the request without S/MIME, if appropriate; however, this 415 response may constitute a downgrade attack.

If a user agent sends an S/MIME body in a request, but receives a response that contains a MIME body that is not secured, the UAC SHOULD notify its user that the session could not be secured. However, if a user agent that supports S/MIME receives a request with an unsecured body, it SHOULD NOT respond with a secured body, but if it expects S/MIME from the sender (for example, because the sender's From header field value corresponds to an identity on its keychain), the UAS SHOULD notify its user that the session could not be secured.

A number of conditions that arise in the previous text call for the notification of the user when an anomalous certificate-management event occurs. Users might well ask what they should do under these circumstances. First and foremost, an unexpected change in a certificate, or an absence of security when security is expected, are





causes for caution but not necessarily indications that an attack is in progress. Users might abort any connection attempt or refuse a connection request they have received; in telephony parlance, they could hang up and call back. Users may wish to find an alternate means to contact the other party and confirm that their key has legitimately changed. Note that users are sometimes compelled to change their certificates, for example when they suspect that the secrecy of their private key has been compromised. When their private key is no longer private, users must legitimately generate a new key and re-establish trust with any users that held their old key.

Finally, if during the course of a dialog a UA receives a certificate in a CMS SignedData message that does not correspond with the certificates previously exchanged during a dialog, the UA MUST notify its user of the change, preferably in terms that indicate that this is a potential security breach.

### 23.3 Securing MIME bodies

There are two types of secure MIME bodies that are of interest to SIP: use of these bodies should follow the S/MIME specification [24] with a few variations.

- "multipart/signed" MUST be used only with CMS detached signatures.

  This allows backwards compatibility with non-S/MIME-compliant recipients.

- S/MIME bodies SHOULD have a Content-Disposition header field, and the value of the "handling" parameter SHOULD be "required."

- If a UAC has no certificate on its keyring associated with the

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 822
Keysight, EX-1045
Page 822

address-of-record to which it wants to send a request, it cannot send an encrypted "application/pkcs7-mime" MIME message. UACs MAY send an initial request such as an OPTIONS message with a CMS detached signature in order to solicit the certificate of the remote side (the signature SHOULD be over a "message/sip" body of the type described in Section 23.4).

Note that future standardization work on S/MIME may define non-certificate based keys.

- Senders of S/MIME bodies SHOULD use the "SMIMECapabilities" (see Section 2.5.2 of [24]) attribute to express their capabilities and preferences for further communications. Note especially that senders MAY use the "preferSignedData"





capability to encourage receivers to respond with CMS SignedData messages (for example, when sending an OPTIONS request as described above).

- S/MIME implementations MUST at a minimum support SHA1 as a digital signature algorithm, and 3DES as an encryption algorithm. All other signature and encryption algorithms MAY be supported. Implementations can negotiate support for these algorithms with the "SMIMECapabilities" attribute.

- Each S/MIME body in a SIP message SHOULD be signed with only one certificate. If a UA receives a message with multiple signatures, the outermost signature should be treated as the single certificate for this body. Parallel signatures SHOULD NOT be used.

The following is an example of an encrypted S/MIME SDP body within a SIP message:

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/pkcs7-mime; smime-type=enveloped-data;
     name=smime.p7m
Content-Disposition: attachment; filename=smime.p7m
   handling=required

**********************************************************
* Content-Type: application/sdp                          *
*                                                        *
* v=0                                                    *
* o=alice 53655765 2353687637 IN IP4 pc33.atlanta.com    *
* s=-                                                    *
* t=0 0                                                  *
* c=IN IP4 pc33.atlanta.com                              *
* m=audio 3456 RTP/AVP 0 1 3 99                          *
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 823
Keysight, EX-1045
Page 823

```
* a=rtpmap:0 PCMU/8000                                  *
*******************************************************
```



### 23.4 SIP Header Privacy and Integrity using S/MIME: Tunneling SIP

As a means of providing some degree of end-to-end authentication, integrity or confidentiality for SIP header fields, S/MIME can encapsulate entire SIP messages within MIME bodies of type "message/sip" and then apply MIME security to these bodies in the same manner as typical SIP bodies. These encapsulated SIP requests and responses do not constitute a separate dialog or transaction, they are a copy of the "outer" message that is used to verify integrity or to supply additional information.

If a UAS receives a request that contains a tunneled "message/sip" S/MIME body, it SHOULD include a tunneled "message/sip" body in the response with the same smime-type.

Any traditional MIME bodies (such as SDP) SHOULD be attached to the "inner" message so that they can also benefit from S/MIME security. Note that "message/sip" bodies can be sent as a part of a MIME "multipart/mixed" body if any unsecured MIME types should also be transmitted in a request.

#### 23.4.1 Integrity and Confidentiality Properties of SIP Headers

When the S/MIME integrity or confidentiality mechanisms are used, there may be discrepancies between the values in the "inner" message and values in the "outer" message. The rules for handling any such differences for all of the header fields described in this document are given in this section.

Note that for the purposes of loose timestamping, all SIP messages that tunnel "message/sip" SHOULD contain a Date header in both the "inner" and "outer" headers.

##### 23.4.1.1 Integrity

Whenever integrity checks are performed, the integrity of a header field should be determined by matching the value of the header field in the signed body with that in the "outer" messages using the comparison rules of SIP as described in 20.

Header fields that can be legitimately modified by proxy servers are: Request-URI, Via, Record-Route, Route, Max-Forwards, and Proxy-Authorization. If these header fields are not intact end-to-end, implementations SHOULD NOT consider this a breach of security. Changes to any other header fields defined in this document constitute an integrity violation; users MUST be notified of a discrepancy.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 824
Keysight, EX-1045
Page 824

23.4.1.2 Confidentiality

When messages are encrypted, header fields may be included in the encrypted body that are not present in the "outer" message.

Some header fields must always have a plaintext version because they are required header fields in requests and responses - these include:

To, From, Call-ID, CSeq, Contact. While it is probably not useful to provide an encrypted alternative for the Call-ID, CSeq, or Contact, providing an alternative to the information in the "outer" To or From is permitted. Note that the values in an encrypted body are not used for the purposes of identifying transactions or dialogs - they are merely informational. If the From header field in an encrypted body differs from the value in the "outer" message, the value within the encrypted body SHOULD be displayed to the user, but MUST NOT be used in the "outer" header fields of any future messages.

Primarily, a user agent will want to encrypt header fields that have an end-to-end semantic, including: Subject, Reply-To, Organization, Accept, Accept-Encoding, Accept-Language, Alert-Info, Error-Info, Authentication-Info, Expires, In-Reply-To, Require, Supported, Unsupported, Retry-After, User-Agent, Server, and Warning. If any of these header fields are present in an encrypted body, they should be used instead of any "outer" header fields, whether this entails displaying the header field values to users or setting internal states in the UA. They SHOULD NOT however be used in the "outer" headers of any future messages.

If present, the Date header field MUST always be the same in the "inner" and "outer" headers.

Since MIME bodies are attached to the "inner" message, implementations will usually encrypt MIME-specific header fields, including: MIME-Version, Content-Type, Content-Length, Content-Language, Content-Encoding and Content-Disposition. The "outer" message will have the proper MIME header fields for S/MIME bodies. These header fields (and any MIME bodies they preface) should be treated as normal MIME header fields and bodies received in a SIP message.

It is not particularly useful to encrypt the following header fields: Min-Expires, Timestamp, Authorization, Priority, and WWW-Authenticate. This category also includes those header fields that can be changed by proxy servers (described in the preceding section). UAs SHOULD never include these in an "inner" message if they are not



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

included in the "outer" message. UAs that receive any of these header fields in an encrypted body SHOULD ignore the encrypted values.

Note that extensions to SIP may define additional header fields; the authors of these extensions should describe the integrity and confidentiality properties of such header fields. If a SIP UA encounters an unknown header field with an integrity violation, it MUST ignore the header field.

### 23.4.2 Tunneling Integrity and Authentication

Tunneling SIP messages within S/MIME bodies can provide integrity for SIP header fields if the header fields that the sender wishes to secure are replicated in a "message/sip" MIME body signed with a CMS detached signature.

Provided that the "message/sip" body contains at least the fundamental dialog identifiers (To, From, Call-ID, CSeq), then a signed MIME body can provide limited authentication. At the very least, if the certificate used to sign the body is unknown to the recipient and cannot be verified, the signature can be used to ascertain that a later request in a dialog was transmitted by the same certificate-holder that initiated the dialog. If the recipient of the signed MIME body has some stronger incentive to trust the certificate (they were able to validate it, they acquired it from a trusted repository, or they have used it frequently) then the signature can be taken as a stronger assertion of the identity of the subject of the certificate.

In order to eliminate possible confusions about the addition or subtraction of entire header fields, senders SHOULD replicate all header fields from the request within the signed body. Any message bodies that require integrity protection MUST be attached to the "inner" message.

If a Date header is present in a message with a signed body, the recipient SHOULD compare the header field value with its own internal clock, if applicable. If a significant time discrepancy is detected (on the order of an hour or more), the user agent SHOULD alert the user to the anomaly, and note that it is a potential security breach.

If an integrity violation in a message is detected by its recipient, the message MAY be rejected with a 403 (Forbidden) response if it is a request, or any existing dialog MAY be terminated. UAs SHOULD notify users of this circumstance and request explicit guidance on how to proceed.





The following is an example of the use of a tunneled "message/sip" body:

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
To: Bob <sip:bob@biloxi.com>
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 826
Keysight, EX-1045
Page 826

```
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Thu, 21 Feb 2002 13:02:03 GMT
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: multipart/signed;
  protocol="application/pkcs7-signature";
  micalg=sha1; boundary=boundary42
Content-Length: 568

--boundary42
Content-Type: message/sip

INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
To: Bob <bob@biloxi.com>
From: Alice <alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Thu, 21 Feb 2002 13:02:03 GMT
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 pc33.atlanta.com
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

--boundary42
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s;
   handling=required
```





```
ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfHfYT6
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
n8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
7GhIGfHfYT64VQbnj756

--boundary42-
```

### 23.4.3 Tunneling Encryption

It may also be desirable to use this mechanism to encrypt a "message/sip" MIME body within a CMS EnvelopedData message S/MIME body, but in practice, most header fields are of at least some use to the network; the general use of encryption with S/MIME is to secure

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 827
Keysight, EX-1045
Page 827

message bodies like SDP rather than message headers. Some informational header fields, such as the Subject or Organization could perhaps warrant end-to-end security. Headers defined by future SIP applications might also require obfuscation.

Another possible application of encrypting header fields is selective anonymity. A request could be constructed with a From header field that contains no personal information (for example, sip:anonymous@anonymizer.invalid). However, a second From header field containing the genuine address-of-record of the originator could be encrypted within a "message/sip" MIME body where it will only be visible to the endpoints of a dialog.

> Note that if this mechanism is used for anonymity, the From header field will no longer be usable by the recipient of a message as an index to their certificate keychain for retrieving the proper S/MIME key to associated with the sender. The message must first be decrypted, and the "inner" From header field MUST be used as an index.

In order to provide end-to-end integrity, encrypted "message/sip" MIME bodies SHOULD be signed by the sender. This creates a "multipart/signed" MIME body that contains an encrypted body and a signature, both of type "application/pkcs7-mime".





In the following example, of an encrypted and signed message, the text boxed in asterisks ("*") is encrypted:

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
To: Bob <sip:bob@biloxi.com>
From: Anonymous <sip:anonymous@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Thu, 21 Feb 2002 13:02:03 GMT
Contact: <sip:pc33.atlanta.com>
Content-Type: multipart/signed;
  protocol="application/pkcs7-signature";
  micalg=sha1; boundary=boundary42
Content-Length: 568

--boundary42
Content-Type: application/pkcs7-mime; smime-type=enveloped-data;
     name=smime.p7m
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

```
  Content-Transfer-Encoding: base64
  Content-Disposition: attachment; filename=smime.p7m
     handling=required
  Content-Length: 231

*************************************************************
* Content-Type: message/sip                                 *
*                                                           *
* INVITE sip:bob@biloxi.com SIP/2.0                         *
* Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8   *
* To: Bob <bob@biloxi.com>                                  *
* From: Alice <alice@atlanta.com>;tag=1928301774            *
* Call-ID: a84b4c76e66710                                   *
* CSeq: 314159 INVITE                                       *
* Max-Forwards: 70                                          *
* Date: Thu, 21 Feb 2002 13:02:03 GMT                       *
* Contact: <sip:alice@pc33.atlanta.com>                     *
*                                                           *
* Content-Type: application/sdp                             *
*                                                           *
* v=0                                                       *
* o=alice 53655765 2353687637 IN IP4 pc33.atlanta.com       *
* s=Session SDP                                             *
* t=0 0                                                     *
* c=IN IP4 pc33.atlanta.com                                 *
* m=audio 3456 RTP/AVP 0 1 3 99                             *
* a=rtpmap:0 PCMU/8000                                      *
*************************************************************
```





```
  --boundary42
  Content-Type: application/pkcs7-signature; name=smime.p7s
  Content-Transfer-Encoding: base64
  Content-Disposition: attachment; filename=smime.p7s;
     handling=required

  ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfHfYT6
  4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
  n8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
  7GhIGfHfYT64VQbnj756

  --boundary42-
```

## 24 Examples

In the following examples, we often omit the message body and the corresponding Content-Length and Content-Type header fields for brevity.

### 24.1 Registration

Bob registers on start-up. The message flow is shown in Figure 9. Note that the authentication usually required for registration is not shown for simplicity.

```
               biloxi.com         Bob's
                registrar       softphone
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 829
Keysight, EX-1045
Page 829

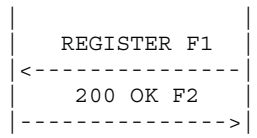



Figure 9: SIP Registration Example

F1 REGISTER Bob -> Registrar

```
REGISTER sip:registrar.biloxi.com SIP/2.0
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Bob <sip:bob@biloxi.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Contact: <sip:bob@192.0.2.4>
Expires: 7200
Content-Length: 0
```





The registration expires after two hours. The registrar responds with a 200 OK:

F2 200 OK Registrar -> Bob

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;branch=z9hG4bKnashds7
 ;received=192.0.2.4
To: Bob <sip:bob@biloxi.com>;tag=2493k59kd
From: Bob <sip:bob@biloxi.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Contact: <sip:bob@192.0.2.4>
Expires: 7200
Content-Length: 0
```

## 24.2 Session Setup

This example contains the full details of the example session setup in Section 4. The message flow is shown in Figure 1. Note that these flows show the minimum required set of header fields - some other header fields such as Allow and Supported would normally be present.

F1 INVITE Alice -> atlanta.com proxy

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

```
Content-Type: application/sdp
Content-Length: 142

(Alice's SDP not shown)
```





F2 100 Trying atlanta.com proxy -> Alice

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Content-Length: 0
```

F3 INVITE atlanta.com proxy -> biloxi.com proxy

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
 ;received=192.0.2.1
Max-Forwards: 69
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142

(Alice's SDP not shown)
```

F4 100 Trying biloxi.com proxy -> atlanta.com proxy

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1
 ;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Content-Length: 0
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 831
Keysight, EX-1045
Page 831

F5 INVITE biloxi.com proxy -> Bob

```
INVITE sip:bob@192.0.2.4 SIP/2.0
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bK4b43c2ff8.1
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1
 ;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
 ;received=192.0.2.1
Max-Forwards: 68
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142

(Alice's SDP not shown)
```

F6 180 Ringing Bob -> biloxi.com proxy

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bK4b43c2ff8.1
 ;received=192.0.2.3
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1
 ;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
Contact: <sip:bob@192.0.2.4>
CSeq: 314159 INVITE
Content-Length: 0
```

F7 180 Ringing biloxi.com proxy -> atlanta.com proxy

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1
 ;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
Contact: <sip:bob@192.0.2.4>
CSeq: 314159 INVITE
Content-Length: 0
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

F8 180 Ringing atlanta.com proxy -> Alice

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
Contact: <sip:bob@192.0.2.4>
CSeq: 314159 INVITE
Content-Length: 0
```

F9 200 OK Bob -> biloxi.com proxy

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bK4b43c2ff8.1
 ;received=192.0.2.3
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1
 ;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:bob@192.0.2.4>
Content-Type: application/sdp
Content-Length: 131

(Bob's SDP not shown)
```

F10 200 OK biloxi.com proxy -> atlanta.com proxy

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1
 ;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:bob@192.0.2.4>
Content-Type: application/sdp
Content-Length: 131

(Bob's SDP not shown)
```



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

```
F11 200 OK atlanta.com proxy -> Alice

SIP/2.0 200 OK
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:bob@192.0.2.4>
Content-Type: application/sdp
Content-Length: 131

(Bob's SDP not shown)

F12 ACK Alice -> Bob

ACK sip:bob@192.0.2.4 SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds9
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 ACK
Content-Length: 0
```

The media session between Alice and Bob is now established.

Bob hangs up first. Note that Bob's SIP phone maintains its own CSeq numbering space, which, in this example, begins with 231. Since Bob is making the request, the To and From URIs and tags have been swapped.

```
F13 BYE Bob -> Alice

BYE sip:alice@pc33.atlanta.com SIP/2.0
Via: SIP/2.0/UDP 192.0.2.4;branch=z9hG4bKnashds10
Max-Forwards: 70
From: Bob <sip:bob@biloxi.com>;tag=a6c85cf
To: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 231 BYE
Content-Length: 0
```



```
F14 200 OK Alice -> Bob

SIP/2.0 200 OK
Via: SIP/2.0/UDP 192.0.2.4;branch=z9hG4bKnashds10
From: Bob <sip:bob@biloxi.com>;tag=a6c85cf
To: Alice <sip:alice@atlanta.com>;tag=1928301774
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 834
Keysight, EX-1045
Page 834

```
Call-ID: a84b4c76e66710
CSeq: 231 BYE
Content-Length: 0
```

The SIP Call Flows document [40] contains further examples of SIP messages.

## 25 Augmented BNF for the SIP Protocol

All of the mechanisms specified in this document are described in both prose and an augmented Backus-Naur Form (BNF) defined in RFC 2234 [10]. Section 6.1 of RFC 2234 defines a set of core rules that are used by this specification, and not repeated here. Implementers need to be familiar with the notation and content of RFC 2234 in order to understand this specification. Certain basic rules are in uppercase, such as SP, LWS, HTAB, CRLF, DIGIT, ALPHA, etc. Angle brackets are used within definitions to clarify the use of rule names.

The use of square brackets is redundant syntactically. It is used as a semantic hint that the specific parameter is optional to use.

### 25.1 Basic Rules

The following rules are used throughout this specification to describe basic parsing constructs. The US-ASCII coded character set is defined by ANSI X3.4-1986.

```
alphanum  =  ALPHA / DIGIT
```





Several rules are incorporated from RFC 2396 [5] but are updated to make them compliant with RFC 2234 [10]. These include:

```
reserved    =  ";" / "/" / "?" / ":" / "@" / "&" / "=" / "+"
               / "$" / ","
unreserved  =  alphanum / mark
mark        =  "-" / "_" / "." / "!" / "~" / "*" / "'"
               / "(" / ")"
escaped     =  "%" HEXDIG HEXDIG
```

SIP header field values can be folded onto multiple lines if the continuation line begins with a space or horizontal tab. All linear white space, including folding, has the same semantics as SP. A

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 835
Keysight, EX-1045
Page 835

recipient MAY replace any linear white space with a single SP before interpreting the field value or forwarding the message downstream. This is intended to behave exactly as HTTP/1.1 as described in RFC 2616 [8]. The SWS construct is used when linear white space is optional, generally between tokens and separators.

```
LWS  =  [*WSP CRLF] 1*WSP ; linear whitespace
SWS  =  [LWS] ; sep whitespace
```

To separate the header name from the rest of value, a colon is used, which, by the above rule, allows whitespace before, but no line break, and whitespace after, including a linebreak. The HCOLON defines this construct.

```
HCOLON  =  *( SP / HTAB ) ":" SWS
```

The TEXT-UTF8 rule is only used for descriptive field contents and values that are not intended to be interpreted by the message parser. Words of *TEXT-UTF8 contain characters from the UTF-8 charset (RFC 2279 [7]). The TEXT-UTF8-TRIM rule is used for descriptive field contents that are n t quoted strings, where leading and trailing LWS is not meaningful. In this regard, SIP differs from HTTP, which uses the ISO 8859-1 character set.

```
TEXT-UTF8-TRIM  =  1*TEXT-UTF8char *(*LWS TEXT-UTF8char)
TEXT-UTF8char   =  %x21-7E / UTF8-NONASCII
UTF8-NONASCII   =  %xC0-DF 1UTF8-CONT
                /  %xE0-EF 2UTF8-CONT
                /  %xF0-F7 3UTF8-CONT
                /  %xF8-Fb 4UTF8-CONT
                /  %xFC-FD 5UTF8-CONT
UTF8-CONT       =  %x80-BF
```





A CRLF is allowed in the definition of TEXT-UTF8-TRIM only as part of a header field continuation. It is expected that the folding LWS will be replaced with a single SP before interpretation of the TEXT-UTF8-TRIM value.

Hexadecimal numeric characters are used in several protocol elements. Some elements (authentication) force hex alphas to be lower case.

```
LHEX  =  DIGIT / %x61-66 ;lowercase a-f
```

Many SIP header field values consist of words separated by LWS or special characters. Unless otherwise stated, tokens are case-insensitive. These special characters MUST be in a quoted string to be used within a parameter value. The word construct is used in Call-ID to allow most separators to be used.

```
token       =  1*(alphanum / "-" / "." / "!" / "%" / "*"
               / "_" / "+" / "`" / "'" / "~" )
separators  =  "(" / ")" / "<" / ">" / "@" /
               "," / ";" / ":" / "\" / DQUOTE /
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 836
Keysight, EX-1045
Page 836

```
                    "/" / "[" / "]" / "?" / "=" /
                    "{" / "}" / SP / HTAB
    word        =   1*(alphanum / "-" / "." / "!" / "%" / "*" /
                    "_" / "+" / "`" / "'" / "~" /
                    "(" / ")" / "<" / ">" /
                    ":" / "\" / DQUOTE /
                    "/" / "[" / "]" / "?" /
                    "{" / "}" )
```

When tokens are used or separators are used between elements, whitespace is often allowed before or after these characters:

```
   STAR    =  SWS "*" SWS ; asterisk
   SLASH   =  SWS "/" SWS ; slash
   EQUAL   =  SWS "=" SWS ; equal
   LPAREN  =  SWS "(" SWS ; left parenthesis
   RPAREN  =  SWS ")" SWS ; right parenthesis
   RAQUOT  =  ">" SWS ; right angle quote
   LAQUOT  =  SWS "<"; left angle quote
   COMMA   =  SWS "," SWS ; comma
   SEMI    =  SWS ";" SWS ; semicolon
   COLON   =  SWS ":" SWS ; colon
   LDQUOT  =  SWS DQUOTE; open double quotation mark
   RDQUOT  =  DQUOTE SWS ; close double quotation mark
```





Comments can be included in some SIP header fields by surrounding the comment text with parentheses. Comments are only allowed in fields containing "comment" as part of their field value definition. In all other fields, parentheses are considered part of the field value.

```
   comment  =  LPAREN *(ctext / quoted-pair / comment) RPAREN
   ctext    =  %x21-27 / %x2A-5B / %x5D-7E / UTF8-NONASCII
               / LWS
```

ctext includes all chars except left and right parens and backslash. A string of text is parsed as a single word if it is quoted using double-quote marks. In quoted strings, quotation marks (") and backslashes (\) need to be escaped.

```
   quoted-string  =  SWS DQUOTE *(qdtext / quoted-pair ) DQUOTE
   qdtext         =  LWS / %x21 / %x23-5B / %x5D-7E
                     / UTF8-NONASCII
```

The backslash character ("\") MAY be used as a single-character quoting mechanism only within quoted-string and comment constructs. Unlike HTTP/1.1, the characters CR and LF cannot be escaped by this mechanism to avoid conflict with line folding and header separation.

```
quoted-pair  =  "\" (%x00-09 / %x0B-0C
              / %x0E-7F)

SIP-URI          =  "sip:" [ userinfo ] hostport
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 837
Keysight, EX-1045
Page 837

```
                    uri-parameters [ headers ]
SIPS-URI         =  "sips:" [ userinfo ] hostport
                    uri-parameters [ headers ]
userinfo         =  ( user / telephone-subscriber ) [ ":" password ] "@"
user             =  1*( unreserved / escaped / user-unreserved )
user-unreserved  =  "&" / "=" / "+" / "$" / "," / ";" / "?" / "/"
password         =  *( unreserved / escaped /
                    "&" / "=" / "+" / "$" / "," )
hostport         =  host [ ":" port ]
host             =  hostname / IPv4address / IPv6reference
hostname         =  *( domainlabel "." ) toplabel [ "." ]
domainlabel      =  alphanum
                    / alphanum *( alphanum / "-" ) alphanum
toplabel         =  ALPHA / ALPHA *( alphanum / "-" ) alphanum
```





```
IPv4address    =  1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT
IPv6reference  =  "[" IPv6address "]"
IPv6address    =  hexpart [ ":" IPv4address ]
hexpart        =  hexseq / hexseq "::" [ hexseq ] / "::" [ hexseq ]
hexseq         =  hex4 *( ":" hex4)
hex4           =  1*4HEXDIG
port           =  1*DIGIT
```

The BNF for telephone-subscriber can be found in RFC 2806 [9]. Note, however, that any characters allowed there that are not allowed in the user part of the SIP URI MUST be escaped.

```
uri-parameters    =  *( ";" uri-parameter)
uri-parameter     =  transport-param / user-param / method-param
                     / ttl-param / maddr-param / lr-param / other-param
transport-param   =  "transport="
                     ( "udp" / "tcp" / "sctp" / "tls"
                     / other-transport)
other-transport   =  token
user-param        =  "user=" ( "phone" / "ip" / other-user)
other-user        =  token
method-param      =  "method=" Method
ttl-param         =  "ttl=" ttl
maddr-param       =  "maddr=" host
lr-param          =  "lr"
other-param       =  pname [ "=" pvalue ]
pname             =  1*paramchar
pvalue            =  1*paramchar
paramchar         =  param-unreserved / unreserved / escaped
param-unreserved  =  "[" / "]" / "/" / ":" / "&" / "+" / "$"

headers         =  "?" header *( "&" header )
header          =  hname "=" hvalue
hname           =  1*( hnv-unreserved / unreserved / escaped )
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 838
Keysight, EX-1045
Page 838

```
hvalue          =  *( hnv-unreserved / unreserved / escaped )
hnv-unreserved  =  "[" / "]" / "/" / "?" / ":" / "+" / "$"

SIP-message     =  Request / Response
Request         =  Request-Line
                   *( message-header )
                   CRLF
                   [ message-body ]
Request-Line    =  Method SP Request-URI SP SIP-Version CRLF
Request-URI     =  SIP-URI / SIPS-URI / absoluteURI
absoluteURI     =  scheme ":" ( hier-part / opaque-part )
hier-part       =  ( net-path / abs-path ) [ "?" query ]
net-path        =  "//" authority [ abs-path ]
abs-path        =  "/" path-segments
```





```
opaque-part     =  uric-no-slash *uric
uric            =  reserved / unreserved / escaped
uric-no-slash   =  unreserved / escaped / ";" / "?" / ":" / "@"
                   / "&" / "=" / "+" / "$" / ","
path-segments   =  segment *( "/" segment )
segment         =  *pchar *( ";" param )
param           =  *pchar
pchar           =  unreserved / escaped /
                   ":" / "@" / "&" / "=" / "+" / "$" / ","
scheme          =  ALPHA *( ALPHA / DIGIT / "+" / "-" / "." )
authority       =  srvr / reg-name
srvr            =  [ [ userinfo "@" ] hostport ]
reg-name        =  1*( unreserved / escaped / "$" / ","
                   / ";" / ":" / "@" / "&" / "=" / "+" )
query           =  *uric
SIP-Version     =  "SIP" "/" 1*DIGIT "." 1*DIGIT

message-header  =  (Accept
                /  Accept-Encoding
                /  Accept-Language
                /  Alert-Info
                /  Allow
                /  Authentication-Info
                /  Authorization
                /  Call-ID
                /  Call-Info
                /  Contact
                /  Content-Disposition
                /  Content-Encoding
                /  Content-Language
                /  Content-Length
                /  Content-Type
                /  CSeq
                /  Date
                /  Error-Info
                /  Expires
                /  From
                /  In-Reply-To
                /  Max-Forwards
                /  MIME-Version
                /  Min-Expires
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

```
                 /  Organization
                 /  Priority
                 /  Proxy-Authenticate
                 /  Proxy-Authorization
                 /  Proxy-Require
                 /  Record-Route
                 /  Reply-To
```





```
                 /  Require
                 /  Retry-After
                 /  Route
                 /  Server
                 /  Subject
                 /  Supported
                 /  Timestamp
                 /  To
                 /  Unsupported
                 /  User-Agent
                 /  Via
                 /  Warning
                 /  WWW-Authenticate
                 /  extension-header) CRLF

INVITEm           =  %x49.4E.56.49.54.45 ; INVITE in caps
ACKm              =  %x41.43.4B ; ACK in caps
OPTIONSm          =  %x4F.50.54.49.4F.4E.53 ; OPTIONS in caps
BYEm              =  %x42.59.45 ; BYE in caps
CANCELm           =  %x43.41.4E.43.45.4C ; CANCEL in caps
REGISTERm         =  %x52.45.47.49.53.54.45.52 ; REGISTER in caps
Method            =  INVITEm / ACKm / OPTIONSm / BYEm
                     / CANCELm / REGISTERm
                     / extension-method
extension-method  =  token
Response          =  Status-Line
                     *( message-header )
                     CRLF
                     [ message-body ]

Status-Line     =  SIP-Version SP Status-Code SP Reason-Phrase CRLF
Status-Code     =  Informational
                /  Redirection
                /  Success
                /  Client-Error
                /  Server-Error
                /  Global-Failure
                /  extension-code
extension-code  =  3DIGIT
Reason-Phrase   =  *(reserved / unreserved / escaped
                   / UTF8-NONASCII / UTF8-CONT / SP / HTAB)

Informational  =  "100"  ;  Trying
               /  "180"  ;  Ringing
               /  "181"  ;  Call Is Being Forwarded
               /  "182"  ;  Queued
               /  "183"  ;  Session Progress
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

```
Success  =  "200"  ;  OK

Redirection  =  "300"  ;  Multiple Choices
             /  "301"  ;  Moved Permanently
             /  "302"  ;  Moved Temporarily
             /  "305"  ;  Use Proxy
             /  "380"  ;  Alternative Service

Client-Error  =  "400"  ;  Bad Request
              /  "401"  ;  Unauthorized
              /  "402"  ;  Payment Required
              /  "403"  ;  Forbidden
              /  "404"  ;  Not Found
              /  "405"  ;  Method Not Allowed
              /  "406"  ;  Not Acceptable
              /  "407"  ;  Proxy Authentication Required
              /  "408"  ;  Request Timeout
              /  "410"  ;  Gone
              /  "413"  ;  Request Entity Too Large
              /  "414"  ;  Request-URI Too Large
              /  "415"  ;  Unsupported Media Type
              /  "416"  ;  Unsupported URI Scheme
              /  "420"  ;  Bad Extension
              /  "421"  ;  Extension Required
              /  "423"  ;  Interval Too Brief
              /  "480"  ;  Temporarily not available
              /  "481"  ;  Call Leg/Transaction Does Not Exist
              /  "482"  ;  Loop Detected
              /  "483"  ;  Too Many Hops
              /  "484"  ;  Address Incomplete
              /  "485"  ;  Ambiguous
              /  "486"  ;  Busy Here
              /  "487"  ;  Request Terminated
              /  "488"  ;  Not Acceptable Here
              /  "491"  ;  Request Pending
              /  "493"  ;  Undecipherable

Server-Error  =  "500"  ;  Internal Server Error
              /  "501"  ;  Not Implemented
              /  "502"  ;  Bad Gateway
              /  "503"  ;  Service Unavailable
              /  "504"  ;  Server Time-out
              /  "505"  ;  SIP Version not supported
              /  "513"  ;  Message Too Large
```



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 841
Keysight, EX-1045
Page 841

```
Global-Failure  =  "600"  ;  Busy Everywhere
               /   "603"  ;  Decline
               /   "604"  ;  Does not exist anywhere
               /   "606"  ;  Not Acceptable

Accept         =  "Accept" HCOLON
                   [ accept-range *(COMMA accept-range) ]
accept-range   =  media-range *(SEMI accept-param)
media-range    =  ( "*/*"
                  / ( m-type SLASH "*" )
                  / ( m-type SLASH m-subtype )
                  ) *( SEMI m-parameter )
accept-param   =  ("q" EQUAL qvalue) / generic-param
qvalue         =  ( "0" [ "." 0*3DIGIT ] )
                  / ( "1" [ "." 0*3("0") ] )
generic-param  =  token [ EQUAL gen-value ]
gen-value      =  token / host / quoted-string

Accept-Encoding  =  "Accept-Encoding" HCOLON
                    [ encoding *(COMMA encoding) ]
encoding         =  codings *(SEMI accept-param)
codings          =  content-coding / "*"
content-coding   =  token

Accept-Language  =  "Accept-Language" HCOLON
                    [ language *(COMMA language) ]
language         =  language-range *(SEMI accept-param)
language-range   =  ( ( 1*8ALPHA *( "-" 1*8ALPHA ) ) / "*" )

Alert-Info   =  "Alert-Info" HCOLON alert-param *(COMMA alert-param)
alert-param  =  LAQUOT absoluteURI RAQUOT *( SEMI generic-param )

Allow  =  "Allow" HCOLON [Method *(COMMA Method)]

Authorization     =  "Authorization" HCOLON credentials
credentials       =  ("Digest" LWS digest-response)
                     / other-response
digest-response   =  dig-resp *(COMMA dig-resp)
dig-resp          =  username / realm / nonce / digest-uri
                      / dresponse / algorithm / cnonce
                      / opaque / message-qop
                      / nonce-count / auth-param
username          =  "username" EQUAL username-value
username-value    =  quoted-string
digest-uri        =  "uri" EQUAL LDQUOT digest-uri-value RDQUOT
digest-uri-value  =  rquest-uri ; Equal to request-uri as specified
                     by HTTP/1.1
message-qop       =  "qop" EQUAL qop-value
```



```
cnonce            =  "cnonce" EQUAL cnonce-value
cnonce-value      =  nonce-value
nonce-count       =  "nc" EQUAL nc-value
nc-value          =  8LHEX
dresponse         =  "response" EQUAL request-digest
request-digest    =  LDQUOT 32LHEX RDQUOT
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 842
Keysight, EX-1045
Page 842

```
auth-param        =  auth-param-name EQUAL
                     ( token / quoted-string )
auth-param-name   =  token
other-response    =  auth-scheme LWS auth-param
                     *(COMMA auth-param)
auth-scheme       =  token

Authentication-Info  =  "Authentication-Info" HCOLON ainfo
                        *(COMMA ainfo)
ainfo                =  nextnonce / message-qop
                         / response-auth / cnonce
                         / nonce-count
nextnonce            =  "nextnonce" EQUAL nonce-value
response-auth        =  "rspauth" EQUAL response-digest
response-digest      =  LDQUOT *LHEX RDQUOT

Call-ID  =  ( "Call-ID" / "i" ) HCOLON callid
callid   =  word [ "@" word ]

Call-Info   =  "Call-Info" HCOLON info *(COMMA info)
info        =  LAQUOT absoluteURI RAQUOT *( SEMI info-param)
info-param  =  ( "purpose" EQUAL ( "icon" / "info"
               / "card" / token ) ) / generic-param

Contact        =  ("Contact" / "m" ) HCOLON
                  ( STAR / (contact-param *(COMMA contact-param)))
contact-param  =  (name-addr / addr-spec) *(SEMI contact-params)
name-addr      =  [ display-name ] LAQUOT addr-spec RAQUOT
addr-spec      =  SIP-URI / SIPS-URI / absoluteURI
display-name   =  *(token LWS)/ quoted-string

contact-params     =  c-p-q / c-p-expires
                      / contact-extension
c-p-q              =  "q" EQUAL qvalue
c-p-expires        =  "expires" EQUAL delta-seconds
contact-extension  =  generic-param
delta-seconds      =  1*DIGIT

Content-Disposition   =  "Content-Disposition" HCOLON
                         disp-type *( SEMI disp-param )
disp-type             =  "render" / "session" / "icon" / "alert"
                         / disp-extension-token
```





```
disp-param            =  handling-param / generic-param
handling-param        =  "handling" EQUAL
                         ( "optional" / "required"
                         / other-handling )
other-handling        =  token
disp-extension-token  =  token

Content-Encoding  =  ( "Content-Encoding" / "e" ) HCOLON
                     content-coding *(COMMA content-coding)

Content-Language  =  "Content-Language" HCOLON
                     language-tag *(COMMA language-tag)
language-tag      =  primary-tag *( "-" subtag )
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 843
Keysight, EX-1045
Page 843

```
primary-tag       =  1*8ALPHA
subtag            =  1*8ALPHA

Content-Length  =  ( "Content-Length" / "l" ) HCOLON 1*DIGIT
Content-Type     =  ( "Content-Type" / "c" ) HCOLON media-type
media-type       =  m-type SLASH m-subtype *(SEMI m-parameter)
m-type           =  discrete-type / composite-type
discrete-type    =  "text" / "image" / "audio" / "video"
                    / "application" / extension-token
composite-type   =  "message" / "multipart" / extension-token
extension-token  =  ietf-token / x-token
ietf-token       =  token
x-token          =  "x-" token
m-subtype        =  extension-token / iana-token
iana-token       =  token
m-parameter      =  m-attribute EQUAL m-value
m-attribute      =  token
m-value          =  token / quoted-string

CSeq  =  "CSeq" HCOLON 1*DIGIT LWS Method

Date          =  "Date" HCOLON SIP-date
SIP-date      =  rfc1123-date
rfc1123-date  =  wkday "," SP date1 SP time SP "GMT"
date1         =  2DIGIT SP month SP 4DIGIT
                 ; day month year (e.g., 02 Jun 1982)
time          =  2DIGIT ":" 2DIGIT ":" 2DIGIT
                 ; 00:00:00 - 23:59:59
wkday         =  "Mon" / "Tue" / "Wed"
                 / "Thu" / "Fri" / "Sat" / "Sun"
month         =  "Jan" / "Feb" / "Mar" / "Apr"
                 / "May" / "Jun" / "Jul" / "Aug"
                 / "Sep" / "Oct" / "Nov" / "Dec"

Error-Info  =  "Error-Info" HCOLON error-uri *(COMMA error-uri)
```





```
error-uri   =  LAQUOT absoluteURI RAQUOT *( SEMI generic-param )

Expires     =  "Expires" HCOLON delta-seconds
From        =  ( "From" / "f" ) HCOLON from-spec
from-spec   =  ( name-addr / addr-spec )
               *( SEMI from-param )
from-param  =  tag-param / generic-param
tag-param   =  "tag" EQUAL token

In-Reply-To  =  "In-Reply-To" HCOLON callid *(COMMA callid)

Max-Forwards  =  "Max-Forwards" HCOLON 1*DIGIT

MIME-Version  =  "MIME-Version" HCOLON 1*DIGIT "." 1*DIGIT

Min-Expires  =  "Min-Expires" HCOLON delta-seconds

Organization  =  "Organization" HCOLON [TEXT-UTF8-TRIM]

Priority        =  "Priority" HCOLON priority-value
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

```
priority-value  =  "emergency" / "urgent" / "normal"
                  / "non-urgent" / other-priority
other-priority  =  token

Proxy-Authenticate  =  "Proxy-Authenticate" HCOLON challenge
challenge           =  ("Digest" LWS digest-cln *(COMMA digest-cln))
                       / other-challenge
other-challenge     =  auth-scheme LWS auth-param
                       *(COMMA auth-param)
digest-cln          =  realm / domain / nonce
                        / opaque / stale / algorithm
                        / qop-options / auth-param
realm               =  "realm" EQUAL realm-value
realm-value         =  quoted-string
domain              =  "domain" EQUAL LDQUOT URI
                       *( 1*SP URI ) RDQUOT
URI                 =  absoluteURI / abs-path
nonce               =  "nonce" EQUAL nonce-value
nonce-value         =  quoted-string
opaque              =  "opaque" EQUAL quoted-string
stale               =  "stale" EQUAL ( "true" / "false" )
algorithm           =  "algorithm" EQUAL ( "MD5" / "MD5-sess"
                       / token )
qop-options         =  "qop" EQUAL LDQUOT qop-value
                       *("," qop-value) RDQUOT
qop-value           =  "auth" / "auth-int" / token

Proxy-Authorization  =  "Proxy-Authorization" HCOLON credentials
```





```
Proxy-Require  =  "Proxy-Require" HCOLON option-tag
                  *(COMMA option-tag)
option-tag     =  token

Record-Route  =  "Record-Route" HCOLON rec-route *(COMMA rec-route)
rec-route     =  name-addr *( SEMI rr-param )
rr-param      =  generic-param

Reply-To      =  "Reply-To" HCOLON rplyto-spec
rplyto-spec   =  ( name-addr / addr-spec )
                 *( SEMI rplyto-param )
rplyto-param  =  generic-param
Require       =  "Require" HCOLON option-tag *(COMMA option-tag)

Retry-After  =  "Retry-After" HCOLON delta-seconds
                [ comment ] *( SEMI retry-param )

retry-param  =  ("duration" EQUAL delta-seconds)
                / generic-param

Route        =  "Route" HCOLON route-param *(COMMA route-param)
route-param  =  name-addr *( SEMI rr-param )

Server           =  "Server" HCOLON server-val *(LWS server-val)
server-val       =  product / comment
product          =  token [SLASH product-version]
product-version  =  token
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

```
Subject  =  ( "Subject" / "s" ) HCOLON [TEXT-UTF8-TRIM]

Supported  =  ( "Supported" / "k" ) HCOLON
              [option-tag *(COMMA option-tag)]

Timestamp  =  "Timestamp" HCOLON 1*(DIGIT)
               [ "." *(DIGIT) ] [ LWS delay ]
delay      =  *(DIGIT) [ "." *(DIGIT) ]

To        =  ( "To" / "t" ) HCOLON ( name-addr
             / addr-spec ) *( SEMI to-param )
to-param  =  tag-param / generic-param

Unsupported  =  "Unsupported" HCOLON option-tag *(COMMA option-tag)
User-Agent  =  "User-Agent" HCOLON server-val *(LWS server-val)
```



```
Via               =  ( "Via" / "v" ) HCOLON via-parm *(COMMA via-parm)
via-parm          =  sent-protocol LWS sent-by *( SEMI via-params )
via-params        =  via-ttl / via-maddr
                     / via-received / via-branch
                     / via-extension
via-ttl           =  "ttl" EQUAL ttl
via-maddr         =  "maddr" EQUAL host
via-received      =  "received" EQUAL (IPv4address / IPv6address)
via-branch        =  "branch" EQUAL token
via-extension     =  generic-param
sent-protocol     =  protocol-name SLASH protocol-version
                     SLASH transport
protocol-name     =  "SIP" / token
protocol-version  =  token
transport         =  "UDP" / "TCP" / "TLS" / "SCTP"
                     / other-transport
sent-by           =  host [ COLON port ]
ttl               =  1*3DIGIT ; 0 to 255

Warning        =  "Warning" HCOLON warning-value *(COMMA warning-value)
warning-value  =  warn-code SP warn-agent SP warn-text
warn-code      =  3DIGIT
warn-agent     =  hostport / pseudonym
                  ;  the name or pseudonym of the server adding
                  ;  the Warning header, for use in debugging
warn-text      =  quoted-string
pseudonym      =  token

WWW-Authenticate  =  "WWW-Authenticate" HCOLON challenge

extension-header  =  header-name HCOLON header-value
header-name       =  token
header-value      =  *(TEXT-UTF8char / UTF8-CONT / LWS)
message-body  =  *OCTET
```

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

# 26 Security Considerations: Threat Model and Security Usage Recommendations

SIP is not an easy protocol to secure. Its use of intermediaries, its multi-faceted trust relationships, its expected usage between elements with no trust at all, and its user-to-user operation make security far from trivial. Security solutions are needed that are deployable today, without extensive coordination, in a wide variety of environments and usages. In order to meet these diverse needs, several distinct mechanisms applicable to different aspects and usages of SIP will be required.





Note that the security of SIP signaling itself has no bearing on the security of protocols used in concert with SIP such as RTP, or with the security implications of any specific bodies SIP might carry (although MIME security plays a substantial role in securing SIP). Any media associated with a session can be encrypted end-to-end independently of any associated SIP signaling. Media encryption is outside the scope of this document.

The considerations that follow first examine a set of classic threat models that broadly identify the security needs of SIP. The set of security services required to address these threats is then detailed, followed by an explanation of several security mechanisms that can be used to provide these services. Next, the requirements for implementers of SIP are enumerated, along with exemplary deployments in which these security mechanisms could be used to improve the security of SIP. Some notes on privacy conclude this section.

## 26.1 Attacks and Threat Models

This section details some threats that should be common to most deployments of SIP. These threats have been chosen specifically to illustrate each of the security services that SIP requires.

The following examples by no means provide an exhaustive list of the threats against SIP; rather, these are "classic" threats that demonstrate the need for particular security services that can potentially prevent whole categories of threats.

These attacks assume an environment in which attackers can potentially read any packet on the network - it is anticipated that SIP will frequently be used on the public Internet. Attackers on the network may be able to modify packets (perhaps at some compromised intermediary). Attackers may wish to steal services, eavesdrop on communications, or disrupt sessions.

### 26.1.1 Registration Hijacking

The SIP registration mechanism allows a user agent to identify itself to a registrar as a device at which a user (designated by an address of record) is located. A registrar assesses the identity asserted in the From header field of a REGISTER message to determine whether this

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 847
Keysight, EX-1045
Page 847

request can modify the contact addresses associated with the address-of-record in the To header field. While these two fields are frequently the same, there are many valid deployments in which a third-party may register contacts on a user's behalf.



The From header field of a SIP request, however, can be modified arbitrarily by the owner of a UA, and this opens the door to malicious registrations. An attacker that successfully impersonates a party authorized to change contacts associated with an address-of-record could, for example, de-register all existing contacts for a URI and then register their own device as the appropriate contact address, thereby directing all requests for the affected user to the attacker's device.

This threat belongs to a family of threats that rely on the absence of cryptographic assurance of a request's originator. Any SIP UAS that represents a valuable service (a gateway that interworks SIP requests with traditional telephone calls, for example) might want to control access to its resources by authenticating requests that it receives. Even end-user UAs, for example SIP phones, have an interest in ascertaining the identities of originators of requests.

This threat demonstrates the need for security services that enable SIP entities to authenticate the originators of requests.

### 26.1.2 Impersonating a Server

The domain to which a request is destined is generally specified in the Request-URI. UAs commonly contact a server in this domain directly in order to deliver a request. However, there is always a possibility that an attacker could impersonate the remote server, and that the UA's request could be intercepted by some other party.

For example, consider a case in which a redirect server at one domain, chicago.com, impersonates a redirect server at another domain, biloxi.com. A user agent sends a request to biloxi.com, but the redirect server at chicago.com answers with a forged response that has appropriate SIP header fields for a response from biloxi.com. The forged contact addresses in the redirection response could direct the originating UA to inappropriate or insecure resources, or simply prevent requests for biloxi.com from succeeding.

This family of threats has a vast membership, many of which are critical. As a converse to the registration hijacking threat, consider the case in which a registration sent to biloxi.com is intercepted by chicago.com, which replies to the intercepted registration with a forged 301 (Moved Permanently) response. This response might seem to come from biloxi.com yet designate chicago.com as the appropriate registrar. All future REGISTER requests from the originating UA would then go to chicago.com.

Prevention of this threat requires a means by which UAs can authenticate the servers to whom they send requests.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

### 26.1.3 Tampering with Message Bodies

As a matter of course, SIP UAs route requests through trusted proxy servers. Regardless of how that trust is established (authentication of proxies is discussed elsewhere in this section), a UA may trust a proxy server to route a request, but not to inspect or possibly modify the bodies contained in that request.

Consider a UA that is using SIP message bodies to communicate session encryption keys for a media session. Although it trusts the proxy server of the domain it is contacting to deliver signaling properly, it may not want the administrators of that domain to be capable of decrypting any subsequent media session. Worse yet, if the proxy server were actively malicious, it could modify the session key, either acting as a man-in-the-middle, or perhaps changing the security characteristics requested by the originating UA.

This family of threats applies not only to session keys, but to most conceivable forms of content carried end-to-end in SIP. These might include MIME bodies that should be rendered to the user, SDP, or encapsulated telephony signals, among others. Attackers might attempt to modify SDP bodies, for example, in order to point RTP media streams to a wiretapping device in order to eavesdrop on subsequent voice communications.

Also note that some header fields in SIP are meaningful end-to-end, for example, Subject. UAs might be protective of these header fields as well as bodies (a malicious intermediary changing the Subject header field might make an important request appear to be spam, for example). However, since many header fields are legitimately inspected or altered by proxy servers as a request is routed, not all header fields should be secured end-to-end.

For these reasons, the UA might want to secure SIP message bodies, and in some limited cases header fields, end-to-end. The security services required for bodies include confidentiality, integrity, and authentication. These end-to-end services should be independent of the means used to secure interactions with intermediaries such as proxy servers.

### 26.1.4 Tearing Down Sessions

Once a dialog has been established by initial messaging, subsequent requests can be sent that modify the state of the dialog and/or session. It is critical that principals in a session can be certain that such requests are not forged by attackers.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 849
Keysight, EX-1045
Page 849

Consider a case in which a third-party attacker captures some initial messages in a dialog shared by two parties in order to learn the parameters of the session (To tag, From tag, and so forth) and then inserts a BYE request into the session. The attacker could opt to forge the request such that it seemed to come from either participant. Once the BYE is received by its target, the session will be torn down prematurely.

Similar mid-session threats include the transmission of forged re-INVITEs that alter the session (possibly to reduce session security or redirect media streams as part of a wiretapping attack).

The most effective countermeasure to this threat is the authentication of the sender of the BYE. In this instance, the recipient needs only know that the BYE came from the same party with whom the corresponding dialog was established (as opposed to ascertaining the absolute identity of the sender). Also, if the attacker is unable to learn the parameters of the session due to confidentiality, it would not be possible to forge the BYE. However, some intermediaries (like proxy servers) will need to inspect those parameters as the session is established.

### 26.1.5 Denial of Service and Amplification

Denial-of-service attacks focus on rendering a particular network element unavailable, usually by directing an excessive amount of network traffic at its interfaces. A distributed denial-of-service attack allows one network user to cause multiple network hosts to flood a target host with a large amount of network traffic.

In many architectures, SIP proxy servers face the public Internet in order to accept requests from worldwide IP endpoints. SIP creates a number of potential opportunities for distributed denial-of-service attacks that must be recognized and addressed by the implementers and operators of SIP systems.

Attackers can create bogus requests that contain a falsified source IP address and a corresponding Via header field that identify a targeted host as the originator of the request and then send this request to a large number of SIP network elements, thereby using hapless SIP UAs or proxies to generate denial-of-service traffic aimed at the target.

Similarly, attackers might use falsified Route header field values in a request that identify the target host and then send such messages to forking proxies that will amplify messaging sent to the target.



Record-Route could be used to similar effect when the attacker is certain that the SIP dialog initiated by the request will result in numerous transactions originating in the backwards direction.

A number of denial-of-service attacks open up if REGISTER requests are not properly authenticated and authorized by registrars.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 850
Keysight, EX-1045
Page 850

Attackers could de-register some or all users in an administrative domain, thereby preventing these users from being invited to new sessions. An attacker could also register a large number of contacts designating the same host for a given address-of-record in order to use the registrar and any associated proxy servers as amplifiers in a denial-of-service attack. Attackers might also attempt to deplete available memory and disk resources of a registrar by registering huge numbers of bindings.

The use of multicast to transmit SIP requests can greatly increase the potential for denial-of-service attacks.

These problems demonstrate a general need to define architectures that minimize the risks of denial-of-service, and the need to be mindful in recommendations for security mechanisms of this class of attacks.

## 26.2 Security Mechanisms

From the threats described above, we gather that the fundamental security services required for the SIP protocol are: preserving the confidentiality and integrity of messaging, preventing replay attacks or message spoofing, providing for the authentication and privacy of the participants in a session, and preventing denial-of-service attacks. Bodies within SIP messages separately require the security services of confidentiality, integrity, and authentication.

Rather than defining new security mechanisms specific to SIP, SIP reuses wherever possible existing security models derived from the HTTP and SMTP space.

Full encryption of messages provides the best means to preserve the confidentiality of signaling - it can also guarantee that messages are not modified by any malicious intermediaries. However, SIP requests and responses cannot be naively encrypted end-to-end in their entirety because message fields such as the Request-URI, Route, and Via need to be visible to proxies in most network architectures so that SIP requests are routed correctly. Note that proxy servers need to modify some features of messages as well (such as adding Via header field values) in order for SIP to function. Proxy servers must therefore be trusted, to some degree, by SIP UAs. To this purpose, low-layer security mechanisms for SIP are recommended, which





encrypt the entire SIP requests or responses on the wire on a hop-by-hop basis, and that allow endpoints to verify the identity of proxy servers to whom they send requests.

SIP entities also have a need to identify one another in a secure fashion. When a SIP endpoint asserts the identity of its user to a peer UA or to a proxy server, that identity should in some way be verifiable. A cryptographic authentication mechanism is provided in SIP to address this requirement.

An independent security mechanism for SIP message bodies supplies an alternative means of end-to-end mutual authentication, as well as providing a limit on the degree to which user agents must trust

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

intermediaries.

### 26.2.1 Transport and Network Layer Security

Transport or network layer security encrypts signaling traffic, guaranteeing message confidentiality and integrity.

Oftentimes, certificates are used in the establishment of lower-layer security, and these certificates can also be used to provide a means of authentication in many architectures.

Two popular alternatives for providing security at the transport and network layer are, respectively, TLS [25] and IPSec [26].

IPSec is a set of network-layer protocol tools that collectively can be used as a secure replacement for traditional IP (Internet Protocol). IPSec is most commonly used in architectures in which a set of hosts or administrative domains have an existing trust relationship with one another. IPSec is usually implemented at the operating system level in a host, or on a security gateway that provides confidentiality and integrity for all traffic it receives from a particular interface (as in a VPN architecture). IPSec can also be used on a hop-by-hop basis.

In many architectures IPSec does not require integration with SIP applications; IPSec is perhaps best suited to deployments in which adding security directly to SIP hosts would be arduous. UAs that have a pre-shared keying relationship with their first-hop proxy server are also good candidates to use IPSec. Any deployment of IPSec for SIP would require an IPSec profile describing the protocol tools that would be required to secure SIP. No such profile is given in this document.





TLS provides transport-layer security over connection-oriented protocols (for the purposes of this document, TCP); "tls" (signifying TLS over TCP) can be specified as the desired transport protocol within a Via header field value or a SIP-URI. TLS is most suited to architectures in which hop-by-hop security is required between hosts with no pre-existing trust association. For example, Alice trusts her local proxy server, which after a certificate exchange decides to trust Bob's local proxy server, which Bob trusts, hence Bob and Alice can communicate securely.

TLS must be tightly coupled with a SIP application. Note that transport mechanisms are specified on a hop-by-hop basis in SIP, thus a UA that sends requests over TLS to a proxy server has no assurance that TLS will be used end-to-end.

The TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite [6] MUST be supported at a minimum by implementers when TLS is used in a SIP application. For purposes of backwards compatibility, proxy servers, redirect servers, and registrars SHOULD support TLS_RSA_WITH_3DES_EDE_CBC_SHA. Implementers MAY also support any other ciphersuite.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 852
Keysight, EX-1045
Page 852

26.2.2 SIPS URI Scheme

The SIPS URI scheme adheres to the syntax of the SIP URI (described in 19), although the scheme string is "sips" rather than "sip". The semantics of SIPS are very different from the SIP URI, however. SIPS allows resources to specify that they should be reached securely.

A SIPS URI can be used as an address-of-record for a particular user - the URI by which the user is canonically known (on their business cards, in the From header field of their requests, in the To header field of REGISTER requests). When used as the Request-URI of a request, the SIPS scheme signifies that each hop over which the request is forwarded, until the request reaches the SIP entity responsible for the domain portion of the Request-URI, must be secured with TLS; once it reaches the domain in question it is handled in accordance with local security and routing policy, quite possibly using TLS for any last hop to a UAS. When used by the originator of a request (as would be the case if they employed a SIPS URI as the address-of-record of the target), SIPS dictates that the entire request path to the target domain be so secured.

The SIPS scheme is applicable to many of the other ways in which SIP URIs are used in SIP today in addition to the Request-URI, including in addresses-of-record, contact addresses (the contents of Contact headers, including those of REGISTER methods), and Route headers. In each instance, the SIPS URI scheme allows these existing fields to





designate secure resources. The manner in which a SIPS URI is dereferenced in any of these contexts has its own security properties which are detailed in [4].

The use of SIPS in particular entails that mutual TLS authentication SHOULD be employed, as SHOULD the ciphersuite TLS_RSA_WITH_AES_128_CBC_SHA. Certificates received in the authentication process SHOULD be validated with root certificates held by the client; failure to validate a certificate SHOULD result in the failure of the request.

> Note that in the SIPS URI scheme, transport is independent of TLS, and thus "sips:alice@atlanta.com;transport=tcp" and "sips:alice@atlanta.com;transport=sctp" are both valid (although note that UDP is not a valid transport for SIPS). The use of "transport=tls" has consequently been deprecated, partly because it was specific to a single hop of the request. This is a change since RFC 2543.

Users that distribute a SIPS URI as an address-of-record may elect to operate devices that refuse requests over insecure transports.

26.2.3 HTTP Authentication

SIP provides a challenge capability, based on HTTP authentication, that relies on the 401 and 407 response codes as well as header fields for carrying challenges and credentials. Without significant

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 853
Keysight, EX-1045
Page 853

modification, the reuse of the HTTP Digest authentication scheme in SIP allows for replay protection and one-way authentication.

The usage of Digest authentication in SIP is detailed in Section 22.

26.2.4 S/MIME

As is discussed above, encrypting entire SIP messages end-to-end for the purpose of confidentiality is not appropriate because network intermediaries (like proxy servers) need to view certain header fields in order to route messages correctly, and if these intermediaries are excluded from security associations, then SIP messages will essentially be non-routable.

However, S/MIME allows SIP UAs to encrypt MIME bodies within SIP, securing these bodies end-to-end without affecting message headers. S/MIME can provide end-to-end confidentiality and integrity for message bodies, as well as mutual authentication. It is also possible to use S/MIME to provide a form of integrity and confidentiality for SIP header fields through SIP message tunneling.





The usage of S/MIME in SIP is detailed in Section 23.

26.3 Implementing Security Mechanisms

26.3.1 Requirements for Implementers of SIP

Proxy servers, redirect servers, and registrars MUST implement TLS, and MUST support both mutual and one-way authentication. It is strongly RECOMMENDED that UAs be capable initiating TLS; UAs MAY also be capable of acting as a TLS server. Proxy servers, redirect servers, and registrars SHOULD possess a site certificate whose subject corresponds to their canonical hostname. UAs MAY have certificates of their own for mutual authentication with TLS, but no provisions are set forth in this document for their use. All SIP elements that support TLS MUST have a mechanism for validating certificates received during TLS negotiation; this entails possession of one or more root certificates issued by certificate authorities (preferably well-known distributors of site certificates comparable to those that issue root certificates for web browsers).

All SIP elements that support TLS MUST also support the SIPS URI scheme.

Proxy servers, redirect servers, registrars, and UAs MAY also implement IPSec or other lower-layer security protocols.

When a UA attempts to contact a proxy server, redirect server, or registrar, the UAC SHOULD initiate a TLS connection over which it will send SIP messages. In some architectures, UASs MAY receive requests over such TLS connections as well.

Proxy servers, redirect servers, registrars, and UAs MUST implement Digest Authorization, encompassing all of the aspects required in 22. Proxy servers, redirect servers, and registrars SHOULD be configured

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 854
Keysight, EX-1045
Page 854

with at least one Digest realm, and at least one "realm" string supported by a given server SHOULD correspond to the server's hostname or domainname.

UAs MAY support the signing and encrypting of MIME bodies, and transference of credentials with S/MIME as described in Section 23. If a UA holds one or more root certificates of certificate authorities in order to validate certificates for TLS or IPSec, it SHOULD be capable of reusing these to verify S/MIME certificates, as appropriate. A UA MAY hold root certificates specifically for validating S/MIME certificates.





Note that is it anticipated that future security extensions may upgrade the normative strength associated with S/MIME as S/MIME implementations appear and the problem space becomes better understood.

### 26.3.2 Security Solutions

The operation of these security mechanisms in concert can follow the existing web and email security models to some degree. At a high level, UAs authenticate themselves to servers (proxy servers, redirect servers, and registrars) with a Digest username and password; servers authenticate themselves to UAs one hop away, or to another server one hop away (and vice versa), with a site certificate delivered by TLS.

On a peer-to-peer level, UAs trust the network to authenticate one another ordinarily; however, S/MIME can also be used to provide direct authentication when the network does not, or if the network itself is not trusted.

The following is an illustrative example in which these security mechanisms are used by various UAs and servers to prevent the sorts of threats described in Section 26.1. While implementers and network administrators MAY follow the normative guidelines given in the remainder of this section, these are provided only as example implementations.

#### 26.3.2.1 Registration

When a UA comes online and registers with its local administrative domain, it SHOULD establish a TLS connection with its registrar (Section 10 describes how the UA reaches its registrar). The registrar SHOULD offer a certificate to the UA, and the site identified by the certificate MUST correspond with the domain in which the UA intends to register; for example, if the UA intends to register the address-of-record 'alice@atlanta.com', the site certificate must identify a host within the atlanta.com domain (such as sip.atlanta.com). When it receives the TLS Certificate message, the UA SHOULD verify the certificate and inspect the site identified by the certificate. If the certificate is invalid, revoked, or if it does not identify the appropriate party, the UA MUST NOT send the

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 855
Keysight, EX-1045
Page 855

REGISTER message and otherwise proceed with the registration.

When a valid certificate has been provided by the registrar, the UA knows that the registrar is not an attacker who might redirect the UA, steal passwords, or attempt any similar attacks.





The UA then creates a REGISTER request that SHOULD be addressed to a Request-URI corresponding to the site certificate received from the registrar. When the UA sends the REGISTER request over the existing TLS connection, the registrar SHOULD challenge the request with a 401 (Proxy Authentication Required) response. The "realm" parameter within the Proxy-Authenticate header field of the response SHOULD correspond to the domain previously given by the site certificate. When the UAC receives the challenge, it SHOULD either prompt the user for credentials or take an appropriate credential from a keyring corresponding to the "realm" parameter in the challenge. The username of this credential SHOULD correspond with the "userinfo" portion of the URI in the To header field of the REGISTER request. Once the Digest credentials have been inserted into an appropriate Proxy-Authorization header field, the REGISTER should be resubmitted to the registrar.

Since the registrar requires the user agent to authenticate itself, it would be difficult for an attacker to forge REGISTER requests for the user's address-of-record. Also note that since the REGISTER is sent over a confidential TLS connection, attackers will not be able to intercept the REGISTER to record credentials for any possible replay attack.

Once the registration has been accepted by the registrar, the UA SHOULD leave this TLS connection open provided that the registrar also acts as the proxy server to which requests are sent for users in this administrative domain. The existing TLS connection will be reused to deliver incoming requests to the UA that has just completed registration.

Because the UA has already authenticated the server on the other side of the TLS connection, all requests that come over this connection are known to have passed through the proxy server - attackers cannot create spoofed requests that appear to have been sent through that proxy server.

#### 26.3.2.2 Interdomain Requests

Now let's say that Alice's UA would like to initiate a session with a user in a remote administrative domain, namely "bob@biloxi.com". We will also say that the local administrative domain (atlanta.com) has a local outbound proxy.

The proxy server that handles inbound requests for an administrative domain MAY also act as a local outbound proxy; for simplicity's sake we'll assume this to be the case for atlanta.com (otherwise the user agent would initiate a new TLS connection to a separate server at this point). Assuming that the client has completed the registration

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 856
Keysight, EX-1045
Page 856

process described in the preceding section, it SHOULD reuse the TLS connection to the local proxy server when it sends an INVITE request to another user. The UA SHOULD reuse cached credentials in the INVITE to avoid prompting the user unnecessarily.

When the local outbound proxy server has validated the credentials presented by the UA in the INVITE, it SHOULD inspect the Request-URI to determine how the message should be routed (see [4]). If the "domainname" portion of the Request-URI had corresponded to the local domain (atlanta.com) rather than biloxi.com, then the proxy server would have consulted its location service to determine how best to reach the requested user.

Had "alice@atlanta.com" been attempting to contact, say, "alex@atlanta.com", the local proxy would have proxied to the request to the TLS connection Alex had established with the registrar when he registered. Since Alex would receive this request over his authenticated channel, he would be assured that Alice's request had been authorized by the proxy server of the local administrative domain.

However, in this instance the Request-URI designates a remote domain. The local outbound proxy server at atlanta.com SHOULD therefore establish a TLS connection with the remote proxy server at biloxi.com. Since both of the participants in this TLS connection are servers that possess site certificates, mutual TLS authentication SHOULD occur. Each side of the connection SHOULD verify and inspect the certificate of the other, noting the domain name that appears in the certificate for comparison with the header fields of SIP messages. The atlanta.com proxy server, for example, SHOULD verify at this stage that the certificate received from the remote side corresponds with the biloxi.com domain. Once it has done so, and TLS negotiation has completed, resulting in a secure channel between the two proxies, the atlanta.com proxy can forward the INVITE request to biloxi.com.

The proxy server at biloxi.com SHOULD inspect the certificate of the proxy server at atlanta.com in turn and compare the domain asserted by the certificate with the "domainname" portion of the From header field in the INVITE request. The biloxi proxy MAY have a strict security policy that requires it to reject requests that do not match the administrative domain from which they have been proxied.

Such security policies could be instituted to prevent the SIP equivalent of SMTP 'open relays' that are frequently exploited to generate spam.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 857
Keysight, EX-1045
Page 857

This policy, however, only guarantees that the request came from the domain it ascribes to itself; it does not allow biloxi.com to ascertain how atlanta.com authenticated Alice. Only if biloxi.com has some other way of knowing atlanta.com's authentication policies could it possibly ascertain how Alice proved her identity. biloxi.com might then institute an even stricter policy that forbids requests that come from domains that are not known administratively to share a common authentication policy with biloxi.com.

Once the INVITE has been approved by the biloxi proxy, the proxy server SHOULD identify the existing TLS channel, if any, associated with the user targeted by this request (in this case "bob@biloxi.com"). The INVITE should be proxied through this channel to Bob. Since the request is received over a TLS connection that had previously been authenticated as the biloxi proxy, Bob knows that the From header field was not tampered with and that atlanta.com has validated Alice, although not necessarily whether or not to trust Alice's identity.

Before they forward the request, both proxy servers SHOULD add a Record-Route header field to the request so that all future requests in this dialog will pass through the proxy servers. The proxy servers can thereby continue to provide security services for the lifetime of this dialog. If the proxy servers do not add themselves to the Record-Route, future messages will pass directly end-to-end between Alice and Bob without any security services (unless the two parties agree on some independent end-to-end security such as S/MIME). In this respect the SIP trapezoid model can provide a nice structure where conventions of agreement between the site proxies can provide a reasonably secure channel between Alice and Bob.

An attacker preying on this architecture would, for example, be unable to forge a BYE request and insert it into the signaling stream between Bob and Alice because the attacker has no way of ascertaining the parameters of the session and also because the integrity mechanism transitively protects the traffic between Alice and Bob.

#### 26.3.2.3 Peer-to-Peer Requests

Alternatively, consider a UA asserting the identity "carol@chicago.com" that has no local outbound proxy. When Carol wishes to send an INVITE to "bob@biloxi.com", her UA SHOULD initiate a TLS connection with the biloxi proxy directly (using the mechanism described in [4] to determine how to best to reach the given Request-URI). When her UA receives a certificate from the biloxi proxy, it SHOULD be verified normally before she passes her INVITE across the TLS connection. However, Carol has no means of proving



her identity to the biloxi proxy, but she does have a CMS-detached signature over a "message/sip" body in the INVITE. It is unlikely in this instance that Carol would have any credentials in the biloxi.com realm, since she has no formal association with biloxi.com. The biloxi proxy MAY also have a strict policy that precludes it from even bothering to challenge requests that do not have biloxi.com in

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 858
Keysight, EX-1045
Page 858

the "domainname" portion of the From header field - it treats these users as unauthenticated.

The biloxi proxy has a policy for Bob that all non-authenticated requests should be redirected to the appropriate contact address registered against 'bob@biloxi.com', namely <sip:bob@192.0.2.4>. Carol receives the redirection response over the TLS connection she established with the biloxi proxy, so she trusts the veracity of the contact address.

Carol SHOULD then establish a TCP connection with the designated address and send a new INVITE with a Request-URI containing the received contact address (recomputing the signature in the body as the request is readied). Bob receives this INVITE on an insecure interface, but his UA inspects and, in this instance, recognizes the From header field of the request and subsequently matches a locally cached certificate with the one presented in the signature of the body of the INVITE. He replies in similar fashion, authenticating himself to Carol, and a secure dialog begins.

> Sometimes firewalls or NATs in an administrative domain could preclude the establishment of a direct TCP connection to a UA. In these cases, proxy servers could also potentially relay requests to UAs in a way that has no trust implications (for example, forgoing an existing TLS connection and forwarding the request over cleartext TCP) as local policy dictates.

26.3.2.4 DoS Protection

In order to minimize the risk of a denial-of-service attack against architectures using these security solutions, implementers should take note of the following guidelines.

When the host on which a SIP proxy server is operating is routable from the public Internet, it SHOULD be deployed in an administrative domain with defensive operational policies (blocking source-routed traffic, preferably filtering ping traffic). Both TLS and IPSec can also make use of bastion hosts at the edges of administrative domains that participate in the security associations to aggregate secure tunnels and sockets. These bastion hosts can also take the brunt of denial-of-service attacks, ensuring that SIP hosts within the administrative domain are not encumbered with superfluous messaging.





No matter what security solutions are deployed, floods of messages directed at proxy servers can lock up proxy server resources and prevent desirable traffic from reaching its destination. There is a computational expense associated with processing a SIP transaction at a proxy server, and that expense is greater for stateful proxy servers than it is for stateless proxy servers. Therefore, stateful proxies are more susceptible to flooding than stateless proxy servers.

UAs and proxy servers SHOULD challenge questionable requests with only a single 401 (Unauthorized) or 407 (Proxy Authentication Required), forgoing the normal response retransmission algorithm, and thus behaving statelessly towards unauthenticated requests.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 859
Keysight, EX-1045
Page 859

Retransmitting the 401 (Unauthorized) or 407 (Proxy Authentication Required) status response amplifies the problem of an attacker using a falsified header field value (such as Via) to direct traffic to a third party.

In summary, the mutual authentication of proxy servers through mechanisms such as TLS significantly reduces the potential for rogue intermediaries to introduce falsified requests or responses that can deny service. This commensurately makes it harder for attackers to make innocent SIP nodes into agents of amplification.

### 26.4 Limitations

Although these security mechanisms, when applied in a judicious manner, can thwart many threats, there are limitations in the scope of the mechanisms that must be understood by implementers and network operators.

#### 26.4.1 HTTP Digest

One of the primary limitations of using HTTP Digest in SIP is that the integrity mechanisms in Digest do not work very well for SIP. Specifically, they offer protection of the Request-URI and the method of a message, but not for any of the header fields that UAs would most likely wish to secure.

The existing replay protection mechanisms described in RFC 2617 also have some limitations for SIP. The next-nonce mechanism, for example, does not support pipelined requests. The nonce-count mechanism should be used for replay protection.

Another limitation of HTTP Digest is the scope of realms. Digest is valuable when a user wants to authenticate themselves to a resource with which they have a pre-existing association, like a service





provider of which the user is a customer (which is quite a common scenario and thus Digest provides an extremely useful function). By way of contrast, the scope of TLS is interdomain or multirealm, since certificates are often globally verifiable, so that the UA can authenticate the server with no pre-existing association.

#### 26.4.2 S/MIME

The largest outstanding defect with the S/MIME mechanism is the lack of a prevalent public key infrastructure for end users. If self-signed certificates (or certificates that cannot be verified by one of the participants in a dialog) are used, the SIP-based key exchange mechanism described in Section 23.2 is susceptible to a man-in-the-middle attack with which an attacker can potentially inspect and modify S/MIME bodies. The attacker needs to intercept the first exchange of keys between the two parties in a dialog, remove the existing CMS-detached signatures from the request and response, and insert a different CMS-detached signature containing a certificate supplied by the attacker (but which seems to be a certificate for the proper address-of-record). Each party will think they have exchanged

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 860
Keysight, EX-1045
Page 860

keys with the other, when in fact each has the public key of the attacker.

It is important to note that the attacker can only leverage this vulnerability on the first exchange of keys between two parties - on subsequent occasions, the alteration of the key would be noticeable to the UAs. It would also be difficult for the attacker to remain in the path of all future dialogs between the two parties over time (as potentially days, weeks, or years pass).

SSH is susceptible to the same man-in-the-middle attack on the first exchange of keys; however, it is widely acknowledged that while SSH is not perfect, it does improve the security of connections. The use of key fingerprints could provide some assistance to SIP, just as it does for SSH. For example, if two parties use SIP to establish a voice communications session, each could read off the fingerprint of the key they received from the other, which could be compared against the original. It would certainly be more difficult for the man-in-the-middle to emulate the voices of the participants than their signaling (a practice that was used with the Clipper chip-based secure telephone).

The S/MIME mechanism allows UAs to send encrypted requests without preamble if they possess a certificate for the destination address-of-record on their keyring. However, it is possible that any particular device registered for an address-of-record will not hold the certificate that has been previously employed by the device's current user, and that it will therefore be unable to process an





encrypted request properly, which could lead to some avoidable error signaling. This is especially likely when an encrypted request is forked.

The keys associated with S/MIME are most useful when associated with a particular user (an address-of-record) rather than a device (a UA). When users move between devices, it may be difficult to transport private keys securely between UAs; how such keys might be acquired by a device is outside the scope of this document.

Another, more prosaic difficulty with the S/MIME mechanism is that it can result in very large messages, especially when the SIP tunneling mechanism described in Section 23.4 is used. For that reason, it is RECOMMENDED that TCP should be used as a transport protocol when S/MIME tunneling is employed.

### 26.4.3 TLS

The most commonly voiced concern about TLS is that it cannot run over UDP; TLS requires a connection-oriented underlying transport protocol, which for the purposes of this document means TCP.

It may also be arduous for a local outbound proxy server and/or registrar to maintain many simultaneous long-lived TLS connections with numerous UAs. This introduces some valid scalability concerns, especially for intensive ciphersuites. Maintaining redundancy of long-lived TLS connections, especially when a UA is solely

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 861
Keysight, EX-1045
Page 861

responsible for their establishment, could also be cumbersome.

TLS only allows SIP entities to authenticate servers to which they are adjacent; TLS offers strictly hop-by-hop security. Neither TLS, nor any other mechanism specified in this document, allows clients to authenticate proxy servers to whom they cannot form a direct TCP connection.

26.4.4 SIPS URIs

Actually using TLS on every segment of a request path entails that the terminating UAS must be reachable over TLS (perhaps registering with a SIPS URI as a contact address). This is the preferred use of SIPS. Many valid architectures, however, use TLS to secure part of the request path, but rely on some other mechanism for the final hop to a UAS, for example. Thus SIPS cannot guarantee that TLS usage will be truly end-to-end. Note that since many UAs will not accept incoming TLS connections, even those UAs that do support TLS may be required to maintain persistent TLS connections as described in the TLS limitations section above in order to receive requests over TLS as a UAS.





Location services are not required to provide a SIPS binding for a SIPS Request-URI. Although location services are commonly populated by user registrations (as described in Section 10.2.1), various other protocols and interfaces could conceivably supply contact addresses for an AOR, and these tools are free to map SIPS URIs to SIP URIs as appropriate. When queried for bindings, a location service returns its contact addresses without regard for whether it received a request with a SIPS Request-URI. If a redirect server is accessing the location service, it is up to the entity that processes the Contact header field of a redirection to determine the propriety of the contact addresses.

Ensuring that TLS will be used for all of the request segments up to the target domain is somewhat complex. It is possible that cryptographically authenticated proxy servers along the way that are non-compliant or compromised may choose to disregard the forwarding rules associated with SIPS (and the general forwarding rules in Section 16.6). Such malicious intermediaries could, for example, retarget a request from a SIPS URI to a SIP URI in an attempt to downgrade security.

Alternatively, an intermediary might legitimately retarget a request from a SIP to a SIPS URI. Recipients of a request whose Request-URI uses the SIPS URI scheme thus cannot assume on the basis of the Request-URI alone that SIPS was used for the entire request path (from the client onwards).

To address these concerns, it is RECOMMENDED that recipients of a request whose Request-URI contains a SIP or SIPS URI inspect the To header field value to see if it contains a SIPS URI (though note that it does not constitute a breach of security if this URI has the same scheme but is not equivalent to the URI in the To header field). Although clients may choose to populate the Request-URI and To header field of a request differently, when SIPS is used this disparity

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 862
Keysight, EX-1045
Page 862

could be interpreted as a possible security violation, and the request could consequently be rejected by its recipient. Recipients MAY also inspect the Via header chain in order to double-check whether or not TLS was used for the entire request path until the local administrative domain was reached. S/MIME may also be used by the originating UAC to help ensure that the original form of the To header field is carried end-to-end.

If the UAS has reason to believe that the scheme of the Request-URI has been improperly modified in transit, the UA SHOULD notify its user of a potential security breach.





As a further measure to prevent downgrade attacks, entities that accept only SIPS requests MAY also refuse connections on insecure ports.

End users will undoubtedly discern the difference between SIPS and SIP URIs, and they may manually edit them in response to stimuli. This can either benefit or degrade security. For example, if an attacker corrupts a DNS cache, inserting a fake record set that effectively removes all SIPS records for a proxy server, then any SIPS requests that traverse this proxy server may fail. When a user, however, sees that repeated calls to a SIPS AOR are failing, they could on some devices manually convert the scheme from SIPS to SIP and retry. Of course, there are some safeguards against this (if the destination UA is truly paranoid it could refuse all non-SIPS requests), but it is a limitation worth noting. On the bright side, users might also divine that 'SIPS' would be valid even when they are presented only with a SIP URI.

### 26.5 Privacy

SIP messages frequently contain sensitive information about their senders - not just what they have to say, but with whom they communicate, when they communicate and for how long, and from where they participate in sessions. Many applications and their users require that this sort of private information be hidden from any parties that do not need to know it.

Note that there are also less direct ways in which private information can be divulged. If a user or service chooses to be reachable at an address that is guessable from the person's name and organizational affiliation (which describes most addresses-of-record), the traditional method of ensuring privacy by having an unlisted "phone number" is compromised. A user location service can infringe on the privacy of the recipient of a session invitation by divulging their specific whereabouts to the caller; an implementation consequently SHOULD be able to restrict, on a per-user basis, what kind of location and availability information is given out to certain classes of callers. This is a whole class of problem that is expected to be studied further in ongoing SIP work.

In some cases, users may want to conceal personal information in

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 863
Keysight, EX-1045
Page 863

header fields that convey identity. This can apply not only to the From and related headers representing the originator of the request, but also the To - it may not be appropriate to convey to the final destination a speed-dialing nickname, or an unexpanded identifier for a group of targets, either of which would be removed from the Request-URI as the request is routed, but not changed in the To





header field if the two were initially identical. Thus it MAY be desirable for privacy reasons to create a To header field that differs from the Request-URI.

## 27 IANA Considerations

All method names, header field names, status codes, and option tags used in SIP applications are registered with IANA through instructions in an IANA Considerations section in an RFC.

The specification instructs the IANA to create four new sub-registries under http://www.iana.org/assignments/sip-parameters: Option Tags, Warning Codes (warn-codes), Methods and Response Codes, added to the sub-registry of Header Fields that is already present there.

### 27.1 Option Tags

This specification establishes the Option Tags sub-registry under http://www.iana.org/assignments/sip-parameters.

Option tags are used in header fields such as Require, Supported, Proxy-Require, and Unsupported in support of SIP compatibility mechanisms for extensions (Section 19.2). The option tag itself is a string that is associated with a particular SIP option (that is, an extension). It identifies the option to SIP endpoints.

Option tags are registered by the IANA when they are published in standards track RFCs. The IANA Considerations section of the RFC must include the following information, which appears in the IANA registry along with the RFC number of the publication.

- Name of the option tag. The name MAY be of any length, but SHOULD be no more than twenty characters long. The name MUST consist of alphanum (Section 25) characters only.

- Descriptive text that describes the extension.

### 27.2 Warn-Codes

This specification establishes the Warn-codes sub-registry under http://www.iana.org/assignments/sip-parameters and initiates its population with the warn-codes listed in Section 20.43. Additional warn-codes are registered by RFC publication.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 864
Keysight, EX-1045
Page 864

The descriptive text for the table of warn-codes is:

Warning codes provide information supplemental to the status code in SIP response messages when the failure of the transaction results from a Session Description Protocol (SDP) (RFC 2327 [1]) problem.

The "warn-code" consists of three digits. A first digit of "3" indicates warnings specific to SIP. Until a future specification describes uses of warn-codes other than 3xx, only 3xx warn-codes may be registered.

Warnings 300 through 329 are reserved for indicating problems with keywords in the session description, 330 through 339 are warnings related to basic network services requested in the session description, 370 through 379 are warnings related to quantitative QoS parameters requested in the session description, and 390 through 399 are miscellaneous warnings that do not fall into one of the above categories.

## 27.3 Header Field Names

This obsoletes the IANA instructions about the header sub-registry under http://www.iana.org/assignments/sip-parameters.

The following information needs to be provided in an RFC publication in order to register a new header field name:

- The RFC number in which the header is registered;
- the name of the header field being registered;
- a compact form version for that header field, if one is defined;

Some common and widely used header fields MAY be assigned one-letter compact forms (Section 7.3.3). Compact forms can only be assigned after SIP working group review, followed by RFC publication.

## 27.4 Method and Response Codes

This specification establishes the Method and Response-Code sub-registries under http://www.iana.org/assignments/sip-parameters and initiates their population as follows. The initial Methods table is:



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

| | |
|---|---|
| INVITE | [RFC3261] |
| ACK | [RFC3261] |
| BYE | [RFC3261] |
| CANCEL | [RFC3261] |
| REGISTER | [RFC3261] |
| OPTIONS | [RFC3261] |
| INFO | [RFC2976] |

The response code table is initially populated from Section 21, the portions labeled Informational, Success, Redirection, Client-Error, Server-Error, and Global-Failure. The table has the following format:

```
Type (e.g., Informational)
      Number    Default Reason Phrase          [RFC3261]
```

The following information needs to be provided in an RFC publication in order to register a new response code or method:

- The RFC number in which the method or response code is registered;
- the number of the response code or name of the method being registered;
- the default reason phrase for that response code, if applicable;

## 27.5 The "message/sip" MIME type.

This document registers the "message/sip" MIME media type in order to allow SIP messages to be tunneled as bodies within SIP, primarily for end-to-end security purposes. This media type is defined by the following information:

```
Media type name: message
Media subtype name: sip
Required parameters: none

Optional parameters: version
   version: The SIP-Version number of the enclosed message (e.g.,
   "2.0"). If not present, the version defaults to "2.0".
Encoding scheme: SIP messages consist of an 8-bit header
   optionally followed by a binary MIME data object. As such, SIP
   messages must be treated as binary. Under normal circumstances
   SIP messages are transported over binary-capable transports, no
   special encodings are needed.
```





```
Security considerations: see below
   Motivation and examples of this usage as a security mechanism
   in concert with S/MIME are given in 23.4.
```

## 27.6 New Content-Disposition Parameter Registrations

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 866
Keysight, EX-1045
Page 866

This document also registers four new Content-Disposition header "disposition-types": alert, icon, session and render. The authors request that these values be recorded in the IANA registry for Content-Dispositions.

Descriptions of these "disposition-types", including motivation and examples, are given in Section 20.11.

Short descriptions suitable for the IANA registry are:

| | |
|---|---|
| alert | the body is a custom ring tone to alert the user |
| icon | the body is displayed as an icon to the user |
| render | the body should be displayed to the user |
| session | the body describes a communications session, for example, as RFC 2327 SDP body |

## 28 Changes From RFC 2543

This RFC revises RFC 2543. It is mostly backwards compatible with RFC 2543. The changes described here fix many errors discovered in RFC 2543 and provide information on scenarios not detailed in RFC 2543. The protocol has been presented in a more cleanly layered model here.

We break the differences into functional behavior that is a substantial change from RFC 2543, which has impact on interoperability or correct operation in some cases, and functional behavior that is different from RFC 2543 but not a potential source of interoperability problems. There have been countless clarifications as well, which are not documented here.

### 28.1 Major Functional Changes

- When a UAC wishes to terminate a call before it has been answered, it sends CANCEL. If the original INVITE still returns a 2xx, the UAC then sends BYE. BYE can only be sent on an existing call leg (now called a dialog in this RFC), whereas it could be sent at any time in RFC 2543.

- The SIP BNF was converted to be RFC 2234 compliant.





- SIP URL BNF was made more general, allowing a greater set of characters in the user part. Furthermore, comparison rules were simplified to be primarily case-insensitive, and detailed handling of comparison in the presence of parameters was described. The most substantial change is that a URI with a parameter with the default value does not match a URI without that parameter.

- Removed Via hiding. It had serious trust issues, since it relied on the next hop to perform the obfuscation process. Instead, Via hiding can be done as a local implementation choice in stateful proxies, and thus is no longer documented.

- In RFC 2543, CANCEL and INVITE transactions were intermingled.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

They are separated now. When a user sends an INVITE and then a CANCEL, the INVITE transaction still terminates normally. A UAS needs to respond to the original INVITE request with a 487 response.

- Similarly, CANCEL and BYE transactions were intermingled; RFC 2543 allowed the UAS not to send a response to INVITE when a BYE was received. That is disallowed here. The original INVITE needs a response.

- In RFC 2543, UAs needed to support only UDP. In this RFC, UAs need to support both UDP and TCP.

- In RFC 2543, a forking proxy only passed up one challenge from downstream elements in the event of multiple challenges. In this RFC, proxies are supposed to collect all challenges and place them into the forwarded response.

- In Digest credentials, the URI needs to be quoted; this is unclear from RFC 2617 and RFC 2069 which are both inconsistent on it.

- SDP processing has been split off into a separate specification [13], and more fully specified as a formal offer/answer exchange process that is effectively tunneled through SIP. SDP is allowed in INVITE/200 or 200/ACK for baseline SIP implementations; RFC 2543 alluded to the ability to use it in INVITE, 200, and ACK in a single transaction, but this was not well specified. More complex SDP usages are allowed in extensions.





- Added full support for IPv6 in URIs and in the Via header field. Support for IPv6 in Via has required that its header field parameters allow the square bracket and colon characters. These characters were previously not permitted. In theory, this could cause interop problems with older implementations. However, we have observed that most implementations accept any non-control ASCII character in these parameters.

- DNS SRV procedure is now documented in a separate specification [4]. This procedure uses both SRV and NAPTR resource records and no longer combines data from across SRV records as described in RFC 2543.

- Loop detection has been made optional, supplanted by a mandatory usage of Max-Forwards. The loop detection procedure in RFC 2543 had a serious bug which would report "spirals" as an error condition when it was not. The optional loop detection procedure is more fully and correctly specified here.

- Usage of tags is now mandatory (they were optional in RFC 2543),

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 868
Keysight, EX-1045
Page 868

as they are now the fundamental building blocks of dialog identification.

- Added the Supported header field, allowing for clients to indicate what extensions are supported to a server, which can apply those extensions to the response, and indicate their usage with a Require in the response.

- Extension parameters were missing from the BNF for several header fields, and they have been added.

- Handling of Route and Record-Route construction was very underspecified in RFC 2543, and also not the right approach. It has been substantially reworked in this specification (and made vastly simpler), and this is arguably the largest change. Backwards compatibility is still provided for deployments that do not use "pre-loaded routes", where the initial request has a set of Route header field values obtained in some way outside of Record-Route. In those situations, the new mechanism is not interoperable.

- In RFC 2543, lines in a message could be terminated with CR, LF, or CRLF. This specification only allows CRLF.





- Usage of Route in CANCEL and ACK was not well defined in RFC 2543. It is now well specified; if a request had a Route header field, its CANCEL or ACK for a non-2xx response to the request need to carry the same Route header field values. ACKs for 2xx responses use the Route values learned from the Record-Route of the 2xx responses.

- RFC 2543 allowed multiple requests in a single UDP packet. This usage has been removed.

- Usage of absolute time in the Expires header field and parameter has been removed. It caused interoperability problems in elements that were not time synchronized, a common occurrence. Relative times are used instead.

- The branch parameter of the Via header field value is now mandatory for all elements to use. It now plays the role of a unique transaction identifier. This avoids the complex and bug-laden transaction identification rules from RFC 2543. A magic cookie is used in the parameter value to determine if the previous hop has made the parameter globally unique, and comparison falls back to the old rules when it is not present. Thus, interoperability is assured.

- In RFC 2543, closure of a TCP connection was made equivalent to a CANCEL. This was nearly impossible to implement (and wrong) for TCP connections between proxies. This has been eliminated, so

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 869
Keysight, EX-1045
Page 869

that there is no coupling between TCP connection state and SIP processing.

- RFC 2543 was silent on whether a UA could initiate a new transaction to a peer while another was in progress. That is now specified here. It is allowed for non-INVITE requests, disallowed for INVITE.

- PGP was removed. It was not sufficiently specified, and not compatible with the more complete PGP MIME. It was replaced with S/MIME.

- Added the "sips" URI scheme for end-to-end TLS. This scheme is not backwards compatible with RFC 2543. Existing elements that receive a request with a SIPS URI scheme in the Request-URI will likely reject the request. This is actually a feature; it ensures that a call to a SIPS URI is only delivered if all path hops can be secured.





- Additional security features were added with TLS, and these are described in a much larger and complete security considerations section.

- In RFC 2543, a proxy was not required to forward provisional responses from 101 to 199 upstream. This was changed to MUST. This is important, since many subsequent features depend on delivery of all provisional responses from 101 to 199.

- Little was said about the 503 response code in RFC 2543. It has since found substantial use in indicating failure or overload conditions in proxies. This requires somewhat special treatment. Specifically, receipt of a 503 should trigger an attempt to contact the next element in the result of a DNS SRV lookup. Also, 503 response is only forwarded upstream by a proxy under certain conditions.

- RFC 2543 defined, but did no sufficiently specify, a mechanism for UA authentication of a server. That has been removed. Instead, the mutual authentication procedures of RFC 2617 are allowed.

- A UA cannot send a BYE for a call until it has received an ACK for the initial INVITE. This was allowed in RFC 2543 but leads to a potential race condition.

- A UA or proxy cannot send CANCEL for a transaction until it gets a provisional response for the request. This was allowed in RFC 2543 but leads to potential race conditions.

- The action parameter in registrations has been deprecated. It was insufficient for any useful services, and caused conflicts when application processing was applied in proxies.

- RFC 2543 had a number of special cases for multicast. For

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 870
Keysight, EX-1045
Page 870

example, certain responses were suppressed, timers were adjusted, and so on. Multicast now plays a more limited role, and the protocol operation is unaffected by usage of multicast as opposed to unicast. The limitations as a result of that are documented.

- Basic authentication has been removed entirely and its usage forbidden.





- Proxies no longer forward a 6xx immediately on receiving it. Instead, they CANCEL pending branches immediately. This avoids a potential race condition that would result in a UAC getting a 6xx followed by a 2xx. In all cases except this race condition, the result will be the same - the 6xx is forwarded upstream.

- RFC 2543 did not address the problem of request merging. This occurs when a request forks at a proxy and later rejoins at an element. Handling of merging is done only at a UA, and procedures are defined for rejecting all but the first request.

28.2 Minor Functional Changes

- Added the Alert-Info, Error-Info, and Call-Info header fields for optional content presentation to users.

- Added the Content-Language, Content-Disposition and MIME-Version header fields.

- Added a "glare handling" mechanism to deal with the case where both parties send each other a re-INVITE simultaneously. It uses the new 491 (Request Pending) error code.

- Added the In-Reply-To and Reply-To header fields for supporting the return of missed calls or messages at a later time.

- Added TLS and SCTP as valid SIP transports.

- There were a variety of mechanisms described for handling failures at any time during a call; those are now generally unified. BYE is sent to terminate.

- RFC 2543 mandated retransmission of INVITE responses over TCP, but noted it was really only needed for 2xx. That was an artifact of insufficient protocol layering. With a more coherent transaction layer defined here, that is no longer needed. Only 2xx responses to INVITEs are retransmitted over TCP.

- Client and server transaction machines are now driven based on timeouts rather than retransmit counts. This allows the state machines to be properly specified for TCP and UDP.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 871
Keysight, EX-1045
Page 871

- The Date header field is used in REGISTER responses to provide a simple means for auto-configuration of dates in user agents.
- Allowed a registrar to reject registrations with expirations that are too short in duration. Defined the 423 response code and the Min-Expires for this purpose.





## 29 Normative References

[1] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

[2] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[3] Resnick, P., "Internet Message Format", RFC 2822, April 2001.

[4] Rosenberg, J. and H. Schulzrinne, "SIP: Locating SIP Servers", RFC 3263, June 2002.

[5] Berners-Lee, T., Fielding, R. and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[6] Chown, P., "Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)", RFC 3268, June 2002.

[7] Yergeau, F., "UTF-8, a transformation format of ISO 10646", RFC 2279, January 1998.

[8] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[9] Vaha-Sipila, A., "URLs for Telephone Calls", RFC 2806, April 2000.

[10] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[11] Freed, F. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

[12] Eastlake, D., Crocker, S. and J. Schiller, "Randomness Recommendations for Security", RFC 1750, December 1994.

[13] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with SDP", RFC 3264, June 2002.

[14] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

[15] Postel, J., "DoD Standard Transmission Control Protocol", RFC 761, January 1980.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 872
Keysight, EX-1045
Page 872

[16] Stewart, R., Xie, Q., Morneault, K., Sharp, C., Schwarzbauer, H., Taylor, T., Rytina, I., Kalla, M., Zhang, L. and V. Paxson, "Stream Control Transmission Protocol", RFC 2960, October 2000.

[17] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A. and L. Stewart, "HTTP authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[18] Troost, R., Dorner, S. and K. Moore, "Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field", RFC 2183, August 1997.

[19] Zimmerer, E., Peterson, J., Vemuri, A., Ong, L., Audet, F., Watson, M. and M. Zonoun, "MIME media types for ISUP and QSIG Objects", RFC 3204, December 2001.

[20] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989.

[21] Alvestrand, H., "IETF Policy on Character Sets and Languages", BCP 18, RFC 2277, January 1998.

[22] Galvin, J., Murphy, S., Crocker, S. and N. Freed, "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", RFC 1847, October 1995.

[23] Housley, R., "Cryptographic Message Syntax", RFC 2630, June 1999.

[24] Ramsdell B., "S/MIME Version 3 Message Specification", RFC 2633, June 1999.

[25] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[26] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

## 30 Informative References

[27] R. Pandya, "Emerging mobile and personal communication systems," IEEE Communications Magazine, Vol. 33, pp. 44--52, June 1995.

[28] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996.



https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 873
Keysight, EX-1045
Page 873

[29] Schulzrinne, H., Rao, R. and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

[30] Cuervo, F., Greene, N., Rayhan, A., Huitema, C., Rosen, B. and J. Segers, "Megaco Protocol Version 1.0", RFC 3015, November 2000.

[31] Handley, M., Schulzrinne, H., Schooler, E. and J. Rosenberg, "SIP: Session Initiation Protocol", RFC 2543, March 1999.

[32] Hoffman, P., Masinter, L. and J. Zawinski, "The mailto URL scheme", RFC 2368, July 1998.

[33] E. M. Schooler, "A multicast user directory service for synchronous rendezvous," Master's Thesis CS-TR-96-18, Department of Computer Science, California Institute of Technology, Pasadena, California, Aug. 1996.

[34] Donovan, S., "The SIP INFO Method", RFC 2976, October 2000.

[35] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

[36] Dawson, F. and T. Howes, "vCard MIME Directory Profile", RFC 2426, September 1998.

[37] Good, G., "The LDAP Data Interchange Format (LDIF) - Technical Specification", RFC 2849, June 2000.

[38] Palme, J., "Common Internet Message Headers", RFC 2076, February 1997.

[39] Franks, J., Hallam-Baker, P., Hostetler, J., Leach, P., Luotonen, A., Sink, E. and L. Stewart, "An Extension to HTTP: Digest Access Authentication", RFC 2069, January 1997.

[40] Johnston, A., Donovan, S., Sparks, R., Cunningham, C., Willis, D., Rosenberg, J., Summers, K. and H. Schulzrinne, "SIP Call Flow Examples", Work in Progress.

[41] E. M. Schooler, "Case study: multimedia conference control in a packet-switched teleconferencing system," Journal of Internetworking: Research and Experience, Vol. 4, pp. 99--120, June 1993. ISI reprint series ISI/RS-93-359.



[42] H. Schulzrinne, "Personal mobility for multimedia services in the Internet," in European Workshop on Interactive Distributed Multimedia Systems and Services (IDMS), (Berlin, Germany), Mar. 1996.

[43] Floyd, S., "Congestion Control Principles", RFC 2914, September

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 874
Keysight, EX-1045
Page 874

2000.





A Table of Timer Values

Table 4 summarizes the meaning and defaults of the various timers used by this specification.

| Timer | Value | Section | Meaning |
|---|---|---|---|
| T1 | 500ms default | Section 17.1.1.1 | RTT Estimate |
| T2 | 4s | Section 17.1.2.2 | The maximum retransmit interval for non-INVITE requests and INVITE responses |
| T4 | 5s | Section 17.1.2.2 | Maximum duration a |

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 875
Keysight, EX-1045
Page 875

| | | | |
|---|---|---|---|
| | | | message will remain in the network |
| Timer A | initially T1 | Section 17.1.1.2 | INVITE request retransmit interval, for UDP only |
| Timer B | 64*T1 | Section 17.1.1.2 | INVITE transaction timeout timer |
| Timer C | > 3min | Section 16.6 bullet 11 | proxy INVITE transaction timeout |
| Timer D | > 32s for UDP 0s for TCP/SCTP | Section 17.1.1.2 | Wait time for response retransmits |
| Timer E | initially T1 | Section 17.1.2.2 | non-INVITE request retransmit interval, UDP only |
| Timer F | 64*T1 | Section 17.1.2.2 | non-INVITE transaction timeout timer |
| Timer G | initially T1 | Section 17.2.1 | INVITE response retransmit interval |
| Timer H | 64*T1 | Section 17.2.1 | Wait time for ACK receipt |
| Timer I | T4 for UDP 0s for TCP/SCTP | Section 17.2.1 | Wait time for ACK retransmits |
| Timer J | 64*T1 for UDP 0s for TCP/SCTP | Section 17.2.2 | Wait time for non-INVITE request retransmits |
| Timer K | T4 for UDP 0s for TCP/SCTP | Section 17.1.2.2 | Wait time for response retransmits |

Table 4: Summary of timers





Acknowledgments

We wish to thank the members of the IETF MMUSIC and SIP WGs for their comments and suggestions. Detailed comments were provided by Ofir Arkin, Brian Bidulock, Jim Buller, Neil Deason, Dave Devanathan, Keith Drage, Bill Fenner, Cedric Fluckiger, Yaron Goland, John Hearty, Bernie Hoeneisen, Jo Hornsby, Phil Hoffer, Christian Huitema, Hisham Khartabil, Jean Jervis, Gadi Karmi, Peter Kjellerstedt, Anders Kristensen, Jonathan Lennox, Gethin Liddell, Allison Mankin, William Marshall, Rohan Mahy, Keith Moore, Vern Paxson, Bob Penfield, Moshe J. Sambol, Chip Sharp, Igor Slepchin, Eric Tremblay, and Rick Workman.

Brian Rosen provided the compiled BNF.

Jean Mahoney provided technical writing assistance.

This work is based, inter alia, on [41,42].

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 876
Keysight, EX-1045
Page 876

Authors' Addresses

Authors addresses are listed alphabetically for the editors, the writers, and then the original authors of RFC 2543. All listed authors actively contributed large amounts of text to this document.

Jonathan Rosenberg
dynamicsoft
72 Eagle Rock Ave
East Hanover, NJ 07936
USA

EMail: jdrosen@dynamicsoft.com

Henning Schulzrinne
Dept. of Computer Science
Columbia University
1214 Amsterdam Avenue
New York, NY 10027
USA

EMail: schulzrinne@cs.columbia.edu

Gonzalo Camarillo
Ericsson

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 877
Keysight, EX-1045
Page 877

Advanced Signalling Research Lab.
FIN-02420 Jorvas
Finland

EMail: Gonzalo.Camarillo@ericsson.com

Alan Johnston
WorldCom
100 South 4th Street
St. Louis, MO 63102
USA

EMail: alan.johnston@wcom.com





Jon Peterson
NeuStar, Inc
1800 Sutter Street, Suite 570
Concord, CA 94520
USA

EMail: jon.peterson@neustar.com

Robert Sparks
dynamicsoft, Inc.
5100 Tennyson Parkway
Suite 1200
Plano, Texas 75024
USA

EMail: rsparks@dynamicsoft.com

Mark Handley
International Computer Science Institute
1947 Center St, Suite 600
Berkeley, CA 94704
USA

EMail: mjh@icir.org

Eve Schooler
AT&T Labs-Research
75 Willow Road
Menlo Park, CA 94025
USA

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

EMail: schooler@research.att.com





Full Copyright Statement

Copyright (C) The Internet Society (2002). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

Html markup produced by rfcmarkup 1.19, available from http://www.levkowetz.com/ietf/tools/rfcmarkup/

https://web.archive.org/web/20060428031957/https://tools.ietf.org/html/rfc3261

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 880
Keysight, EX-1045
Page 880

Network Working Group
Request for Comments: 4566
Obsoletes: 2327, 3266
Category: Standards Track

M. Handley
UCL
V. Jacobson
Packet Design
C. Perkins
University of Glasgow

July 2006

# SDP: Session Description Protocol

## Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Copyright Notice

Copyright (C) The Internet Society (2006).

## Abstract

This memo defines the Session Description Protocol (SDP). SDP is intended for describing multimedia sessions for the purposes of session announcement, session invitation, and other forms of multimedia session initiation.

## Table of Contents

1. Introduction ....3
2. Glossary of Terms ....3
3. Examples of SDP Usage ....4
   3.1. Session Initiation ....4
   3.2. Streaming Media ....4
   3.3. Email and the World Wide Web ....4
   3.4. Multicast Session Announcement ....4
4. Requirements and Recommendations ....5
   4.1. Media and Transport Information ....6
   4.2. Timing Information ....6
   4.3. Private Sessions ....7
   4.4. Obtaining Further Information about a Session ....7
   4.5. Categorisation ....7
   4.6. Internationalisation ....7



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 881
Keysight, EX-1045
Page 881

5. SDP Specification .....7
5.1. Protocol Version ("v=") .....10
5.2. Origin ("o=") .....11
5.3. Session Name ("s=") .....12
5.4. Session Information ("i=") .....12
5.5. URI ("u=") .....13
5.6. Email Address and Phone Number ("e=" and "p=") .....13
5.7. Connection Data ("c=") .....14
5.8. Bandwidth ("b=") .....16
5.9. Timing ("t=") .....17
5.10. Repeat Times ("r=") .....18
5.11. Time Zones ("z=") .....19
5.12. Encryption Keys ("k=") .....19
5.13. Attributes ("a=") .....21
5.14. Media Descriptions ("m=") .....22
6. SDP Attributes .....24
7. Security Considerations .....31
8. IANA Considerations .....33
8.1. The "application/sdp" Media Type .....33
8.2. Registration of Parameters .....34
8.2.1. Media Types ("media") .....34
8.2.2. Transport Protocols ("proto") .....34
8.2.3. Media Formats ("fmt") .....35
8.2.4. Attribute Names ("att-field") .....36
8.2.5. Bandwidth Specifiers ("bwtype") .....37
8.2.6. Network Types ("nettype") .....37
8.2.7. Address Types ("addrtype") .....38
8.2.8. Registration Procedure .....38
8.3. Encryption Key Access Methods .....39
9. SDP Grammar .....39
10. Summary of Changes from RFC 2327 .....44
11. Acknowledgements .....45
12. References .....45
12.1. Normative References .....45
12.2. Informative References .....46



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 882
Keysight, EX-1045
Page 882

## 1. Introduction

When initiating multimedia teleconferences, voice-over-IP calls, streaming video, or other sessions, there is a requirement to convey media details, transport addresses, and other session description metadata to the participants.

SDP provides a standard representation for such information, irrespective of how that information is transported. SDP is purely a format for session description -- it does not incorporate a transport protocol, and it is intended to use different transport protocols as appropriate, including the Session Announcement Protocol [14], Session Initiation Protocol [15], Real Time Streaming Protocol [16], electronic mail using the MIME extensions, and the Hypertext Transport Protocol.

SDP is intended to be general purpose so that it can be used in a wide range of network environments and applications. However, it is not intended to support negotiation of session content or media encodings: this is viewed as outside the scope of session description.

This memo obsoletes RFC 2327 [6] and RFC 3266 [10]. Section 10 outlines the changes introduced in this memo.

## 2. Glossary of Terms

The following terms are used in this document and have specific meaning within the context of this document.

Conference: A multimedia conference is a set of two or more communicating users along with the software they are using to communicate.

Session: A multimedia session is a set of multimedia senders and receivers and the data streams flowing from senders to receivers. A multimedia conference is an example of a multimedia session.

Session Description: A well-defined format for conveying sufficient information to discover and participate in a multimedia session.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [3].



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 883
Keysight, EX-1045
Page 883

## 3. Examples of SDP Usage

### 3.1. Session Initiation

The Session Initiation Protocol (SIP) [15] is an application-layer control protocol for creating, modifying, and terminating sessions such as Internet multimedia conferences, Internet telephone calls, and multimedia distribution. The SIP messages used to create sessions carry session descriptions that allow participants to agree on a set of compatible media types. These session descriptions are commonly formatted using SDP. When used with SIP, the offer/answer model [17] provides a limited framework for negotiation using SDP.

### 3.2. Streaming Media

The Real Time Streaming Protocol (RTSP) [16], is an application-level protocol for control over the delivery of data with real-time properties. RTSP provides an extensible framework to enable controlled, on-demand delivery of real-time data, such as audio and video. An RTSP client and server negotiate an appropriate set of parameters for media delivery, partially using SDP syntax to describe those parameters.

### 3.3. Email and the World Wide Web

Alternative means of conveying session descriptions include electronic mail and the World Wide Web (WWW). For both email and WWW distribution, the media type "application/sdp" is used. This enables the automatic launching of applications for participation in the session from the WWW client or mail reader in a standard manner.

Note that announcements of multicast sessions made only via email or the WWW do not have the property that the receiver of a session announcement can necessarily receive the session because the multicast sessions may be restricted in scope, and access to the WWW server or reception of email is possible outside this scope.

### 3.4. Multicast Session Announcement

In order to assist the advertisement of multicast multimedia conferences and other multicast sessions, and to communicate the relevant session setup information to prospective participants, a distributed session directory may be used. An instance of such a session directory periodically sends packets containing a description of the session to a well-known multicast group. These advertisements are received by other session directories such that potential remote participants can use the session description to start the tools required to participate in the session.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 884
Keysight, EX-1045
Page 884

One protocol used to implement such a distributed directory is the Session Announcement Protocol (SAP) [14]. SDP provides the recommended session description format for such session announcements.

## 4. Requirements and Recommendations

The purpose of SDP is to convey information about media streams in multimedia sessions to allow the recipients of a session description to participate in the session. SDP is primarily intended for use in an internetwork, although it is sufficiently general that it can describe conferences in other network environments. Media streams can be many-to-many. Sessions need not be continually active.

Thus far, multicast-based sessions on the Internet have differed from many other forms of conferencing in that anyone receiving the traffic can join the session (unless the session traffic is encrypted). In such an environment, SDP serves two primary purposes. It is a means to communicate the existence of a session, and it is a means to convey sufficient information to enable joining and participating in the session. In a unicast environment, only the latter purpose is likely to be relevant.

An SDP session description includes the following:

- Session name and purpose
- Time(s) the session is active
- The media comprising the session
- Information needed to receive those media (addresses, ports, formats, etc.)

As resources necessary to participate in a session may be limited, some additional information may also be desirable:

- Information about the bandwidth to be used by the session
- Contact information for the person responsible for the session

In general, SDP must convey sufficient information to enable applications to join a session (with the possible exception of encryption keys) and to announce the resources to be used to any non-participants that may need to know. (This latter feature is primarily useful when SDP is used with a multicast session announcement protocol.)



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 885
Keysight, EX-1045
Page 885

### 4.1. Media and Transport Information

An SDP session description includes the following media information:

- The type of media (video, audio, etc.)
- The transport protocol (RTP/UDP/IP, H.320, etc.)
- The format of the media (H.261 video, MPEG video, etc.)

In addition to media format and transport protocol, SDP conveys address and port details. For an IP multicast session, these comprise:

- The multicast group address for media
- The transport port for media

This address and port are the destination address and destination port of the multicast stream, whether being sent, received, or both.

For unicast IP sessions, the following are conveyed:

- The remote address for media
- The remote transport port for media

The semantics of this address and port depend on the media and transport protocol defined. By default, this SHOULD be the remote address and remote port to which data is sent. Some media types may redefine this behaviour, but this is NOT RECOMMENDED since it complicates implementations (including middleboxes that must parse the addresses to open Network Address Translation (NAT) or firewall pinholes).

### 4.2. Timing Information

Sessions may be either bounded or unbounded in time. Whether or not they are bounded, they may be only active at specific times. SDP can convey:

- An arbitrary list of start and stop times bounding the session
- For each bound, repeat times such as "every Wednesday at 10am for one hour"

This timing information is globally consistent, irrespective of local time zone or daylight saving time (see Section 5.9).



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 886
Keysight, EX-1045
Page 886

### 4.3. Private Sessions

It is possible to create both public sessions and private sessions. SDP itself does not distinguish between these; private sessions are typically conveyed by encrypting the session description during distribution. The details of how encryption is performed are dependent on the mechanism used to convey SDP; mechanisms are currently defined for SDP transported using SAP [14] and SIP [15], and others may be defined in the future.

If a session announcement is private, it is possible to use that private announcement to convey encryption keys necessary to decode each of the media in a conference, including enough information to know which encryption scheme is used for each media.

### 4.4. Obtaining Further Information about a Session

A session description should convey enough information to decide whether or not to participate in a session. SDP may include additional pointers in the form of Uniform Resource Identifiers (URIs) for more information about the session.

### 4.5. Categorisation

When many session descriptions are being distributed by SAP, or any other advertisement mechanism, it may be desirable to filter session announcements that are of interest from those that are not. SDP supports a categorisation mechanism for sessions that is capable of being automated (the "a=cat:" attribute; see Section 6).

### 4.6. Internationalisation

The SDP specification recommends the use of the ISO 10646 character sets in the UTF-8 encoding [5] to allow many different languages to be represented. However, to assist in compact representations, SDP also allows other character sets such as ISO 8859-1 to be used when desired. Internationalisation only applies to free-text fields (session name and background information), and not to SDP as a whole.

## 5. SDP Specification

An SDP session description is denoted by the media type "application/sdp" (See Section 8).

An SDP session description is entirely textual using the ISO 10646 character set in UTF-8 encoding. SDP field names and attribute names use only the US-ASCII subset of UTF-8, but textual fields and attribute values MAY use the full ISO 10646 character set. Field and



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 887
Keysight, EX-1045
Page 887

attribute values that use the full UTF-8 character set are never directly compared, hence there is no requirement for UTF-8 normalisation. The textual form, as opposed to a binary encoding such as ASN.1 or XDR, was chosen to enhance portability, to enable a variety of transports to be used, and to allow flexible, text-based toolkits to be used to generate and process session descriptions. However, since SDP may be used in environments where the maximum permissible size of a session description is limited, the encoding is deliberately compact. Also, since announcements may be transported via very unreliable means or damaged by an intermediate caching server, the encoding was designed with strict order and formatting rules so that most errors would result in malformed session announcements that could be detected easily and discarded. This also allows rapid discarding of encrypted session announcements for which a receiver does not have the correct key.

An SDP session description consists of a number of lines of text of the form:

```
<type>=<value>
```

where <type> MUST be exactly one case-significant character and <value> is structured text whose format depends on <type>. In general, <value> is either a number of fields delimited by a single space character or a free format string, and is case-significant unless a specific field defines otherwise. Whitespace MUST NOT be used on either side of the "=" sign.

An SDP session description consists of a session-level section followed by zero or more media-level sections. The session-level part starts with a "v=" line and continues to the first media-level section. Each media-level section starts with an "m=" line and continues to the next media-level section or end of the whole session description. In general, session-level values are the default for all media unless overridden by an equivalent media-level value.

Some lines in each description are REQUIRED and some are OPTIONAL, but all MUST appear in exactly the order given here (the fixed order greatly enhances error detection and allows for a simple parser). OPTIONAL items are marked with a "*".



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 888
Keysight, EX-1045
Page 888

```
      Session description
         v=  (protocol version)
         o=  (originator and session identifier)
         s=  (session name)
         i=* (session information)
         u=* (URI of description)
         e=* (email address)
         p=* (phone number)
         c=* (connection information -- not required if included in
              all media)
         b=* (zero or more bandwidth information lines)
         One or more time descriptions ("t=" and "r=" lines; see below)
         z=* (time zone adjustments)
         k=* (encryption key)
         a=* (zero or more session attribute lines)
         Zero or more media descriptions

      Time description
         t=  (time the session is active)
         r=* (zero or more repeat times)

      Media description, if present
         m=  (media name and transport address)
         i=* (media title)
         c=* (connection information -- optional if included at
              session level)
         b=* (zero or more bandwidth information lines)
         k=* (encryption key)
         a=* (zero or more media attribute lines)
```

The set of type letters is deliberately small and not intended to be extensible -- an SDP parser MUST completely ignore any session description that contains a type letter that it does not understand. The attribute mechanism ("a=" described below) is the primary means for extending SDP and tailoring it to particular applications or media. Some attributes (the ones listed in Section 6 of this memo) have a defined meaning, but others may be added on an application-, media-, or session-specific basis. An SDP parser MUST ignore any attribute it doesn't understand.

An SDP session description may contain URIs that reference external content in the "u=", "k=", and "a=" lines. These URIs may be dereferenced in some cases, making the session description non-self-contained.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 889
Keysight, EX-1045
Page 889

The connection ("c=") and attribute ("a=") information in the session-level section applies to all the media of that session unless overridden by connection information or an attribute of the same name in the media description. For instance, in the example below, each media behaves as if it were given a "recvonly" attribute.

An example SDP description is:

```
v=0
o=jdoe 2890844526 2890842807 IN IP4 10.47.16.5
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.example.com/seminars/sdp.pdf
e=j.doe@example.com (Jane Doe)
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 99
a=rtpmap:99 h263-1998/90000
```

Text fields such as the session name and information are octet strings that may contain any octet with the exceptions of 0x00 (Nul), 0x0a (ASCII newline), and 0x0d (ASCII carriage return). The sequence CRLF (0x0d0a) is used to end a record, although parsers SHOULD be tolerant and also accept records terminated with a single newline character. If the "a=charset" attribute is not present, these octet strings MUST be interpreted as containing ISO-10646 characters in UTF-8 encoding (the presence of the "a=charset" attribute may force some fields to be interpreted differently).

A session description can contain domain names in the "o=", "u=", "e=", "c=", and "a=" lines. Any domain name used in SDP MUST comply with [1], [2]. Internationalised domain names (IDNs) MUST be represented using the ASCII Compatible Encoding (ACE) form defined in [11] and MUST NOT be directly represented in UTF-8 or any other encoding (this requirement is for compatibility with RFC 2327 and other SDP-related standards, which predate the development of internationalised domain names).

### 5.1. Protocol Version ("v=")

```
v=0
```

The "v=" field gives the version of the Session Description Protocol. This memo defines version 0. There is no minor version number.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 890
Keysight, EX-1045
Page 890

### 5.2. Origin ("o=")

```
o=<username> <sess-id> <sess-version> <nettype> <addrtype>
  <unicast-address>
```

The "o=" field gives the originator of the session (her username and the address of the user's host) plus a session identifier and version number:

<username> is the user's login on the originating host, or it is "-" if the originating host does not support the concept of user IDs. The <username> MUST NOT contain spaces.

<sess-id> is a numeric string such that the tuple of <username>, <sess-id>, <nettype>, <addrtype>, and <unicast-address> forms a globally unique identifier for the session. The method of <sess-id> allocation is up to the creating tool, but it has been suggested that a Network Time Protocol (NTP) format timestamp be used to ensure uniqueness [13].

<sess-version> is a version number for this session description. Its usage is up to the creating tool, so long as <sess-version> is increased when a modification is made to the session data. Again, it is RECOMMENDED that an NTP format timestamp is used.

<nettype> is a text string giving the type of network. Initially "IN" is defined to have the meaning "Internet", but other values MAY be registered in the future (see Section 8).

<addrtype> is a text string giving the type of the address that follows. Initially "IP4" and "IP6" are defined, but other values MAY be registered in the future (see Section 8).

<unicast-address> is the address of the machine from which the session was created. For an address type of IP4, this is either the fully qualified domain name of the machine or the dotted-decimal representation of the IP version 4 address of the machine. For an address type of IP6, this is either the fully qualified domain name of the machine or the compressed textual representation of the IP version 6 address of the machine. For both IP4 and IP6, the fully qualified domain name is the form that SHOULD be given unless this is unavailable, in which case the globally unique address MAY be substituted. A local IP address MUST NOT be used in any context where the SDP description might leave the scope in which the address is meaningful (for example, a local address MUST NOT be included in an application-level referral that might leave the scope).



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 891
Keysight, EX-1045
Page 891

In general, the "o=" field serves as a globally unique identifier for this version of this session description, and the subfields excepting the version taken together identify the session irrespective of any modifications.

For privacy reasons, it is sometimes desirable to obfuscate the username and IP address of the session originator. If this is a concern, an arbitrary <username> and private <unicast-address> MAY be chosen to populate the "o=" field, provided that these are selected in a manner that does not affect the global uniqueness of the field.

### 5.3. Session Name ("s=")

s=<session name>

The "s=" field is the textual session name. There MUST be one and only one "s=" field per session description. The "s=" field MUST NOT be empty and SHOULD contain ISO 10646 characters (but see also the "a=charset" attribute). If a session has no meaningful name, the value "s= " SHOULD be used (i.e., a single space as the session name).

### 5.4. Session Information ("i=")

i=<session description>

The "i=" field provides textual information about the session. There MUST be at most one session-level "i=" field per session description, and at most one "i=" field per media. If the "a=charset" attribute is present, it specifies the character set used in the "i=" field. If the "a=charset" attribute is not present, the "i=" field MUST contain ISO 10646 characters in UTF-8 encoding.

A single "i=" field MAY also be used for each media definition. In media definitions, "i=" fields are primarily intended for labelling media streams. As such, they are most likely to be useful when a single session has more than one distinct media stream of the same media type. An example would be two different whiteboards, one for slides and one for feedback and questions.

The "i=" field is intended to provide a free-form human-readable description of the session or the purpose of a media stream. It is not suitable for parsing by automata.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 892
Keysight, EX-1045
Page 892

### 5.5. URI ("u=")

```
u=<uri>
```

A URI is a Uniform Resource Identifier as used by WWW clients [7]. The URI should be a pointer to additional information about the session. This field is OPTIONAL, but if it is present it MUST be specified before the first media field. No more than one URI field is allowed per session description.

### 5.6. Email Address and Phone Number ("e=" and "p=")

```
e=<email-address>
p=<phone-number>
```

The "e=" and "p=" lines specify contact information for the person responsible for the conference. This is not necessarily the same person that created the conference announcement.

Inclusion of an email address or phone number is OPTIONAL. Note that the previous version of SDP specified that either an email field or a phone field MUST be specified, but this was widely ignored. The change brings the specification into line with common usage.

If an email address or phone number is present, it MUST be specified before the first media field. More than one email or phone field can be given for a session description.

Phone numbers SHOULD be given in the form of an international public telecommunication number (see ITU-T Recommendation E.164) preceded by a "+". Spaces and hyphens may be used to split up a phone field to aid readability if desired. For example:

```
p=+1 617 555-6011
```

Both email addresses and phone numbers can have an OPTIONAL free text string associated with them, normally giving the name of the person who may be contacted. This MUST be enclosed in parentheses if it is present. For example:

```
e=j.doe@example.com (Jane Doe)
```

The alternative RFC 2822 [29] name quoting convention is also allowed for both email addresses and phone numbers. For example:

```
e=Jane Doe <j.doe@example.com>
```



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 893
Keysight, EX-1045
Page 893

The free text string SHOULD be in the ISO-10646 character set with UTF-8 encoding, or alternatively in ISO-8859-1 or other encodings if the appropriate session-level "a=charset" attribute is set.

### 5.7. Connection Data ("c=")

```
c=<nettype> <addrtype> <connection-address>
```

The "c=" field contains connection data.

A session description MUST contain either at least one "c=" field in each media description or a single "c=" field at the session level. It MAY contain a single session-level "c=" field and additional "c=" field(s) per media description, in which case the per-media values override the session-level settings for the respective media.

The first sub-field ("<nettype>") is the network type, which is a text string giving the type of network. Initially, "IN" is defined to have the meaning "Internet", but other values MAY be registered in the future (see Section 8).

The second sub-field ("<addrtype>") is the address type. This allows SDP to be used for sessions that are not IP based. This memo only defines IP4 and IP6, but other values MAY be registered in the future (see Section 8).

The third sub-field ("<connection-address>") is the connection address. OPTIONAL sub-fields MAY be added after the connection address depending on the value of the <addrtype> field.

When the <addrtype> is IP4 and IP6, the connection address is defined as follows:

- If the session is multicast, the connection address will be an IP multicast group address. If the session is not multicast, then the connection address contains the unicast IP address of the expected data source or data relay or data sink as determined by additional attribute fields. It is not expected that unicast addresses will be given in a session description that is communicated by a multicast announcement, though this is not prohibited.

- Sessions using an IPv4 multicast connection address MUST also have a time to live (TTL) value present in addition to the multicast address. The TTL and the address together define the scope with which multicast packets sent in this conference will be sent. TTL values MUST be in the range 0-255. Although the TTL MUST be specified, its use to scope multicast traffic is deprecated;



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 894
Keysight, EX-1045
Page 894

applications SHOULD use an administratively scoped address instead.

The TTL for the session is appended to the address using a slash as a separator. An example is:

```
c=IN IP4 224.2.36.42/127
```

IPv6 multicast does not use TTL scoping, and hence the TTL value MUST NOT be present for IPv6 multicast. It is expected that IPv6 scoped addresses will be used to limit the scope of conferences.

Hierarchical or layered encoding schemes are data streams where the encoding from a single media source is split into a number of layers. The receiver can choose the desired quality (and hence bandwidth) by only subscribing to a subset of these layers. Such layered encodings are normally transmitted in multiple multicast groups to allow multicast pruning. This technique keeps unwanted traffic from sites only requiring certain levels of the hierarchy. For applications requiring multiple multicast groups, we allow the following notation to be used for the connection address:

```
<base multicast address>[/<ttl>]/<number of addresses>
```

If the number of addresses is not given, it is assumed to be one. Multicast addresses so assigned are contiguously allocated above the base address, so that, for example:

```
c=IN IP4 224.2.1.1/127/3
```

would state that addresses 224.2.1.1, 224.2.1.2, and 224.2.1.3 are to be used at a TTL of 127. This is semantically identical to including multiple "c=" lines in a media description:

```
c=IN IP4 224.2.1.1/127
c=IN IP4 224.2.1.2/127
c=IN IP4 224.2.1.3/127
```

Similarly, an IPv6 example would be:

```
c=IN IP6 FF15::101/3
```

which is semantically equivalent to:

```
c=IN IP6 FF15::101
c=IN IP6 FF15::102
c=IN IP6 FF15::103
```



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 895
Keysight, EX-1045
Page 895

(remembering that the TTL field is not present in IPv6 multicast).

Multiple addresses or "c=" lines MAY be specified on a per-media basis only if they provide multicast addresses for different layers in a hierarchical or layered encoding scheme. They MUST NOT be specified for a session-level "c=" field.

The slash notation for multiple addresses described above MUST NOT be used for IP unicast addresses.

### 5.8. Bandwidth ("b=")

b=<bwtype>:<bandwidth>

This OPTIONAL field denotes the proposed bandwidth to be used by the session or media. The <bwtype> is an alphanumeric modifier giving the meaning of the <bandwidth> figure. Two values are defined in this specification, but other values MAY be registered in the future (see Section 8 and [21], [25]):

CT If the bandwidth of a session or media in a session is different from the bandwidth implicit from the scope, a "b=CT:..." line SHOULD be supplied for the session giving the proposed upper limit to the bandwidth used (the "conference total" bandwidth). The primary purpose of this is to give an approximate idea as to whether two or more sessions can coexist simultaneously. When using the CT modifier with RTP, if several RTP sessions are part of the conference, the conference total refers to total bandwidth of all RTP sessions.

AS The bandwidth is interpreted to be application specific (it will be the application's concept of maximum bandwidth). Normally, this will coincide with what is set on the application's "maximum bandwidth" control if applicable. For RTP-based applications, AS gives the RTP "session bandwidth" as defined in Section 6.2 of [19].

Note that CT gives a total bandwidth figure for all the media at all sites. AS gives a bandwidth figure for a single media at a single site, although there may be many sites sending simultaneously.

A prefix "X-" is defined for <bwtype> names. This is intended for experimental purposes only. For example:

b=X-YZ:128



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 896
Keysight, EX-1045
Page 896

Use of the "X-" prefix is NOT RECOMMENDED: instead new modifiers SHOULD be registered with IANA in the standard namespace. SDP parsers MUST ignore bandwidth fields with unknown modifiers. Modifiers MUST be alphanumeric and, although no length limit is given, it is recommended that they be short.

The <bandwidth> is interpreted as kilobits per second by default. The definition of a new <bwtype> modifier MAY specify that the bandwidth is to be interpreted in some alternative unit (the "CT" and "AS" modifiers defined in this memo use the default units).

### 5.9. Timing ("t=")

```
t=<start-time> <stop-time>
```

The "t=" lines specify the start and stop times for a session. Multiple "t=" lines MAY be used if a session is active at multiple irregularly spaced times; each additional "t=" line specifies an additional period of time for which the session will be active. If the session is active at regular times, an "r=" line (see below) should be used in addition to, and following, a "t=" line -- in which case the "t=" line specifies the start and stop times of the repeat sequence.

The first and second sub-fields give the start and stop times, respectively, for the session. These values are the decimal representation of Network Time Protocol (NTP) time values in seconds since 1900 [13]. To convert these values to UNIX time, subtract decimal 2208988800.

NTP timestamps are elsewhere represented by 64-bit values, which wrap sometime in the year 2036. Since SDP uses an arbitrary length decimal representation, this should not cause an issue (SDP timestamps MUST continue counting seconds since 1900, NTP will use the value modulo the 64-bit limit).

If the <stop-time> is set to zero, then the session is not bounded, though it will not become active until after the <start-time>. If the <start-time> is also zero, the session is regarded as permanent.

User interfaces SHOULD strongly discourage the creation of unbounded and permanent sessions as they give no information about when the session is actually going to terminate, and so make scheduling difficult.

The general assumption may be made, when displaying unbounded sessions that have not timed out to the user, that an unbounded session will only be active until half an hour from the current time



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 897
Keysight, EX-1045
Page 897

or the session start time, whichever is the later. If behaviour other than this is required, an end-time SHOULD be given and modified as appropriate when new information becomes available about when the session should really end.

Permanent sessions may be shown to the user as never being active unless there are associated repeat times that state precisely when the session will be active.

### 5.10. Repeat Times ("r=")

```
r=<repeat interval> <active duration> <offsets from start-time>
```

"r=" fields specify repeat times for a session. For example, if a session is active at 10am on Monday and 11am on Tuesday for one hour each week for three months, then the <start-time> in the corresponding "t=" field would be the NTP representation of 10am on the first Monday, the <repeat interval> would be 1 week, the <active duration> would be 1 hour, and the offsets would be zero and 25 hours. The corresponding "t=" field stop time would be the NTP representation of the end of the last session three months later. By default, all fields are in seconds, so the "r=" and "t=" fields might be the following:

```
t=3034423619 3042462419
r=604800 3600 0 90000
```

To make description more compact, times may also be given in units of days, hours, or minutes. The syntax for these is a number immediately followed by a single case-sensitive character. Fractional units are not allowed -- a smaller unit should be used instead. The following unit specification characters are allowed:

```
d - days (86400 seconds)
h - hours (3600 seconds)
m - minutes (60 seconds)
s - seconds (allowed for completeness)
```

Thus, the above session announcement could also have been written:

```
r=7d 1h 0 25h
```

Monthly and yearly repeats cannot be directly specified with a single SDP repeat time; instead, separate "t=" fields should be used to explicitly list the session times.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 898
Keysight, EX-1045
Page 898

### 5.11. Time Zones ("z=")

```
z=<adjustment time> <offset> <adjustment time> <offset> ....
```

To schedule a repeated session that spans a change from daylight saving time to standard time or vice versa, it is necessary to specify offsets from the base time. This is required because different time zones change time at different times of day, different countries change to or from daylight saving time on different dates, and some countries do not have daylight saving time at all.

Thus, in order to schedule a session that is at the same time winter and summer, it must be possible to specify unambiguously by whose time zone a session is scheduled. To simplify this task for receivers, we allow the sender to specify the NTP time that a time zone adjustment happens and the offset from the time when the session was first scheduled. The "z=" field allows the sender to specify a list of these adjustment times and offsets from the base time.

An example might be the following:

```
z=2882844526 -1h 2898848070 0
```

This specifies that at time 2882844526, the time base by which the session's repeat times are calculated is shifted back by 1 hour, and that at time 2898848070, the session's original time base is restored. Adjustments are always relative to the specified start time -- they are not cumulative. Adjustments apply to all "t=" and "r=" lines in a session description.

If a session is likely to last several years, it is expected that the session announcement will be modified periodically rather than transmit several years' worth of adjustments in one session announcement.

### 5.12. Encryption Keys ("k=")

```
k=<method>
k=<method>:<encryption key>
```

If transported over a secure and trusted channel, the Session Description Protocol MAY be used to convey encryption keys. A simple mechanism for key exchange is provided by the key field ("k="), although this is primarily supported for compatibility with older implementations and its use is NOT RECOMMENDED. Work is in progress to define new key exchange mechanisms for use with SDP [27] [28], and it is expected that new applications will use those mechanisms.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 899
Keysight, EX-1045
Page 899

A key field is permitted before the first media entry (in which case it applies to all media in the session), or for each media entry as required. The format of keys and their usage are outside the scope of this document, and the key field provides no way to indicate the encryption algorithm to be used, key type, or other information about the key: this is assumed to be provided by the higher-level protocol using SDP. If there is a need to convey this information within SDP, the extensions mentioned previously SHOULD be used. Many security protocols require two keys: one for confidentiality, another for integrity. This specification does not support transfer of two keys.

The method indicates the mechanism to be used to obtain a usable key by external means, or from the encoded encryption key given. The following methods are defined:

k=clear:<encryption key>

The encryption key is included untransformed in this key field. This method MUST NOT be used unless it can be guaranteed that the SDP is conveyed over a secure channel. The encryption key is interpreted as text according to the charset attribute; use the "k=base64:" method to convey characters that are otherwise prohibited in SDP.

k=base64:<encoded encryption key>

The encryption key is included in this key field but has been base64 encoded [12] because it includes characters that are prohibited in SDP. This method MUST NOT be used unless it can be guaranteed that the SDP is conveyed over a secure channel.

k=uri:<URI to obtain key>

A Uniform Resource Identifier is included in the key field. The URI refers to the data containing the key, and may require additional authentication before the key can be returned. When a request is made to the given URI, the reply should specify the encoding for the key. The URI is often an Secure Socket Layer/Transport Layer Security (SSL/TLS)-protected HTTP URI ("https:"), although this is not required.

k=prompt

No key is included in this SDP description, but the session or media stream referred to by this key field is encrypted. The user should be prompted for the key when attempting to join the session, and this user-supplied key should then be used to



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 900
Keysight, EX-1045
Page 900

decrypt the media streams. The use of user-specified keys is NOT RECOMMENDED, since such keys tend to have weak security properties.

The key field MUST NOT be used unless it can be guaranteed that the SDP is conveyed over a secure and trusted channel. An example of such a channel might be SDP embedded inside an S/MIME message or a TLS-protected HTTP session. It is important to ensure that the secure channel is with the party that is authorised to join the session, not an intermediary: if a caching proxy server is used, it is important to ensure that the proxy is either trusted or unable to access the SDP.

### 5.13. Attributes ("a=")

```
a=<attribute>
a=<attribute>:<value>
```

Attributes are the primary means for extending SDP. Attributes may be defined to be used as "session-level" attributes, "media-level" attributes, or both.

A media description may have any number of attributes ("a=" fields) that are media specific. These are referred to as "media-level" attributes and add information about the media stream. Attribute fields can also be added before the first media field; these "session-level" attributes convey additional information that applies to the conference as a whole rather than to individual media.

Attribute fields may be of two forms:

- A property attribute is simply of the form "a=<flag>". These are binary attributes, and the presence of the attribute conveys that the attribute is a property of the session. An example might be "a=recvonly".

- A value attribute is of the form "a=<attribute>:<value>". For example, a whiteboard could have the value attribute "a=orient: landscape"

Attribute interpretation depends on the media tool being invoked. Thus receivers of session descriptions should be configurable in their interpretation of session descriptions in general and of attributes in particular.

Attribute names MUST use the US-ASCII subset of ISO-10646/UTF-8.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 901
Keysight, EX-1045
Page 901

Attribute values are octet strings, and MAY use any octet value except 0x00 (Nul), 0x0A (LF), and 0x0D (CR). By default, attribute values are to be interpreted as in ISO-10646 character set with UTF-8 encoding. Unlike other text fields, attribute values are NOT normally affected by the "charset" attribute as this would make comparisons against known values problematic. However, when an attribute is defined, it can be defined to be charset dependent, in which case its value should be interpreted in the session charset rather than in ISO-10646.

Attributes MUST be registered with IANA (see Section 8). If an attribute is received that is not understood, it MUST be ignored by the receiver.

### 5.14. Media Descriptions ("m=")

```
m=<media> <port> <proto> <fmt> ...
```

A session description may contain a number of media descriptions. Each media description starts with an "m=" field and is terminated by either the next "m=" field or by the end of the session description. A media field has several sub-fields:

<media> is the media type. Currently defined media are "audio", "video", "text", "application", and "message", although this list may be extended in the future (see Section 8).

<port> is the transport port to which the media stream is sent. The meaning of the transport port depends on the network being used as specified in the relevant "c=" field, and on the transport protocol defined in the <proto> sub-field of the media field. Other ports used by the media application (such as the RTP Control Protocol (RTCP) port [19]) MAY be derived algorithmically from the base media port or MAY be specified in a separate attribute (for example, "a=rtcp:" as defined in [22]).

If non-contiguous ports are used or if they don't follow the parity rule of even RTP ports and odd RTCP ports, the "a=rtcp:" attribute MUST be used. Applications that are requested to send media to a <port> that is odd and where the "a=rtcp:" is present MUST NOT subtract 1 from the RTP port: that is, they MUST send the RTP to the port indicated in <port> and send the RTCP to the port indicated in the "a=rtcp" attribute.

For applications where hierarchically encoded streams are being sent to a unicast address, it may be necessary to specify multiple transport ports. This is done using a similar notation to that used for IP multicast addresses in the "c=" field:



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 902
Keysight, EX-1045
Page 902

```
   m=<media> <port>/<number of ports> <proto> <fmt> ...
```

In such a case, the ports used depend on the transport protocol. For RTP, the default is that only the even-numbered ports are used for data with the corresponding one-higher odd ports used for the RTCP belonging to the RTP session, and the <number of ports> denoting the number of RTP sessions. For example:

```
   m=video 49170/2 RTP/AVP 31
```

would specify that ports 49170 and 49171 form one RTP/RTCP pair and 49172 and 49173 form the second RTP/RTCP pair. RTP/AVP is the transport protocol and 31 is the format (see below). If non-contiguous ports are required, they must be signalled using a separate attribute (for example, "a=rtcp:" as defined in [22]).

If multiple addresses are specified in the "c=" field and multiple ports are specified in the "m=" field, a one-to-one mapping from port to the corresponding address is implied. For example:

```
   c=IN IP4 224.2.1.1/127/2
   m=video 49170/2 RTP/AVP 31
```

would imply that address 224.2.1.1 is used with ports 49170 and 49171, and address 224.2.1.2 is used with ports 49172 and 49173.

The semantics of multiple "m=" lines using the same transport address are undefined. This implies that, unlike limited past practice, there is no implicit grouping defined by such means and an explicit grouping framework (for example, [18]) should instead be used to express the intended semantics.

<proto> is the transport protocol. The meaning of the transport protocol is dependent on the address type field in the relevant "c=" field. Thus a "c=" field of IP4 indicates that the transport protocol runs over IP4. The following transport protocols are defined, but may be extended through registration of new protocols with IANA (see Section 8):

* udp: denotes an unspecified protocol running over UDP.

* RTP/AVP: denotes RTP [19] used under the RTP Profile for Audio and Video Conferences with Minimal Control [20] running over UDP.

* RTP/SAVP: denotes the Secure Real-time Transport Protocol [23] running over UDP.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 903
Keysight, EX-1045
Page 903

The main reason to specify the transport protocol in addition to the media format is that the same standard media formats may be carried over different transport protocols even when the network protocol is the same -- a historical example is vat Pulse Code Modulation (PCM) audio and RTP PCM audio; another might be TCP/RTP PCM audio. In addition, relays and monitoring tools that are transport-protocol-specific but format-independent are possible.

<fmt> is a media format description. The fourth and any subsequent sub-fields describe the format of the media. The interpretation of the media format depends on the value of the <proto> sub-field.

If the <proto> sub-field is "RTP/AVP" or "RTP/SAVP" the <fmt> sub-fields contain RTP payload type numbers. When a list of payload type numbers is given, this implies that all of these payload formats MAY be used in the session, but the first of these formats SHOULD be used as the default format for the session. For dynamic payload type assignments the "a=rtpmap:" attribute (see Section 6) SHOULD be used to map from an RTP payload type number to a media encoding name that identifies the payload format. The "a=fmtp:" attribute MAY be used to specify format parameters (see Section 6).

If the <proto> sub-field is "udp" the <fmt> sub-fields MUST reference a media type describing the format under the "audio", "video", "text", "application", or "message" top-level media types. The media type registration SHOULD define the packet format for use with UDP transport.

For media using other transport protocols, the <fmt> field is protocol specific. Rules for interpretation of the <fmt> sub-field MUST be defined when registering new protocols (see Section 8.2.2).

## 6. SDP Attributes

The following attributes are defined. Since application writers may add new attributes as they are required, this list is not exhaustive. Registration procedures for new attributes are defined in Section 8.2.4.

a=cat:<category>

This attribute gives the dot-separated hierarchical category of the session. This is to enable a receiver to filter unwanted sessions by category. There is no central registry of categories. It is a session-level attribute, and it is not dependent on charset.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 904
Keysight, EX-1045
Page 904

a=keywds:<keywords>

Like the cat attribute, this is to assist identifying wanted sessions at the receiver. This allows a receiver to select interesting session based on keywords describing the purpose of the session; there is no central registry of keywords. It is a session-level attribute. It is a charset-dependent attribute, meaning that its value should be interpreted in the charset specified for the session description if one is specified, or by default in ISO 10646/UTF-8.

a=tool:<name and version of tool>

This gives the name and version number of the tool used to create the session description. It is a session-level attribute, and it is not dependent on charset.

a=ptime:<packet time>

This gives the length of time in milliseconds represented by the media in a packet. This is probably only meaningful for audio data, but may be used with other media types if it makes sense. It should not be necessary to know ptime to decode RTP or vat audio, and it is intended as a recommendation for the encoding/packetisation of audio. It is a media-level attribute, and it is not dependent on charset.

a=maxptime:<maximum packet time>

This gives the maximum amount of media that can be encapsulated in each packet, expressed as time in milliseconds. The time SHALL be calculated as the sum of the time the media present in the packet represents. For frame-based codecs, the time SHOULD be an integer multiple of the frame size. This attribute is probably only meaningful for audio data, but may be used with other media types if it makes sense. It is a media-level attribute, and it is not dependent on charset. Note that this attribute was introduced after RFC 2327, and non-updated implementations will ignore this attribute.

a=rtpmap:<payload type> <encoding name>/<clock rate> [/<encoding parameters>]

This attribute maps from an RTP payload type number (as used in an "m=" line) to an encoding name denoting the payload format to be used. It also provides information on the clock rate and encoding parameters. It is a media-level attribute that is not dependent on charset.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 905
Keysight, EX-1045
Page 905

Although an RTP profile may make static assignments of payload type numbers to payload formats, it is more common for that assignment to be done dynamically using "a=rtpmap:" attributes. As an example of a static payload type, consider u-law PCM coded single-channel audio sampled at 8 kHz. This is completely defined in the RTP Audio/Video profile as payload type 0, so there is no need for an "a=rtpmap:" attribute, and the media for such a stream sent to UDP port 49232 can be specified as:

```
m=audio 49232 RTP/AVP 0
```

An example of a dynamic payload type is 16-bit linear encoded stereo audio sampled at 16 kHz. If we wish to use the dynamic RTP/AVP payload type 98 for this stream, additional information is required to decode it:

```
m=audio 49232 RTP/AVP 98
a=rtpmap:98 L16/16000/2
```

Up to one rtpmap attribute can be defined for each media format specified. Thus, we might have the following:

```
m=audio 49230 RTP/AVP 96 97 98
a=rtpmap:96 L8/8000
a=rtpmap:97 L16/8000
a=rtpmap:98 L16/11025/2
```

RTP profiles that specify the use of dynamic payload types MUST define the set of valid encoding names and/or a means to register encoding names if that profile is to be used with SDP. The "RTP/AVP" and "RTP/SAVP" profiles use media subtypes for encoding names, under the top-level media type denoted in the "m=" line. In the example above, the media types are "audio/l8" and "audio/l16".

For audio streams, <encoding parameters> indicates the number of audio channels. This parameter is OPTIONAL and may be omitted if the number of channels is one, provided that no additional parameters are needed.

For video streams, no encoding parameters are currently specified.

Additional encoding parameters MAY be defined in the future, but codec-specific parameters SHOULD NOT be added. Parameters added to an "a=rtpmap:" attribute SHOULD only be those required for a session directory to make the choice of appropriate media



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 906
Keysight, EX-1045
Page 906

to participate in a session. Codec-specific parameters should be added in other attributes (for example, "a=fmtp:").

Note: RTP audio formats typically do not include information about the number of samples per packet. If a non-default (as defined in the RTP Audio/Video Profile) packetisation is required, the "ptime" attribute is used as given above.

a=recvonly

This specifies that the tools should be started in receive-only mode where applicable. It can be either a session- or media-level attribute, and it is not dependent on charset. Note that recvonly applies to the media only, not to any associated control protocol (e.g., an RTP-based system in recvonly mode SHOULD still send RTCP packets).

a=sendrecv

This specifies that the tools should be started in send and receive mode. This is necessary for interactive conferences with tools that default to receive-only mode. It can be either a session or media-level attribute, and it is not dependent on charset.

If none of the attributes "sendonly", "recvonly", "inactive", and "sendrecv" is present, "sendrecv" SHOULD be assumed as the default for sessions that are not of the conference type "broadcast" or "H332" (see below).

a=sendonly

This specifies that the tools should be started in send-only mode. An example may be where a different unicast address is to be used for a traffic destination than for a traffic source. In such a case, two media descriptions may be used, one sendonly and one recvonly. It can be either a session- or media-level attribute, but would normally only be used as a media attribute. It is not dependent on charset. Note that sendonly applies only to the media, and any associated control protocol (e.g., RTCP) SHOULD still be received and processed as normal.

a=inactive

This specifies that the tools should be started in inactive mode. This is necessary for interactive conferences where users can put other users on hold. No media is sent over an



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 907
Keysight, EX-1045
Page 907

inactive media stream. Note that an RTP-based system SHOULD still send RTCP, even if started inactive. It can be either a session or media-level attribute, and it is not dependent on charset.

a=orient:<orientation>

Normally this is only used for a whiteboard or presentation tool. It specifies the orientation of a the workspace on the screen. It is a media-level attribute. Permitted values are "portrait", "landscape", and "seascape" (upside-down landscape). It is not dependent on charset.

a=type:<conference type>

This specifies the type of the conference. Suggested values are "broadcast", "meeting", "moderated", "test", and "H332". "recvonly" should be the default for "type:broadcast" sessions, "type:meeting" should imply "sendrecv", and "type:moderated" should indicate the use of a floor control tool and that the media tools are started so as to mute new sites joining the conference.

Specifying the attribute "type:H332" indicates that this loosely coupled session is part of an H.332 session as defined in the ITU H.332 specification [26]. Media tools should be started "recvonly".

Specifying the attribute "type:test" is suggested as a hint that, unless explicitly requested otherwise, receivers can safely avoid displaying this session description to users.

The type attribute is a session-level attribute, and it is not dependent on charset.

a=charset:<character set>

This specifies the character set to be used to display the session name and information data. By default, the ISO-10646 character set in UTF-8 encoding is used. If a more compact representation is required, other character sets may be used. For example, the ISO 8859-1 is specified with the following SDP attribute:

```
a=charset:ISO-8859-1
```



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 908
Keysight, EX-1045
Page 908

This is a session-level attribute and is not dependent on charset. The charset specified MUST be one of those registered with IANA, such as ISO-8859-1. The character set identifier is a US-ASCII string and MUST be compared against the IANA identifiers using a case-insensitive comparison. If the identifier is not recognised or not supported, all strings that are affected by it SHOULD be regarded as octet strings.

Note that a character set specified MUST still prohibit the use of bytes 0x00 (Nul), 0x0A (LF), and 0x0d (CR). Character sets requiring the use of these characters MUST define a quoting mechanism that prevents these bytes from appearing within text fields.

a=sdplang:<language tag>

This can be a session-level attribute or a media-level attribute. As a session-level attribute, it specifies the language for the session description. As a media-level attribute, it specifies the language for any media-level SDP information field associated with that media. Multiple sdplang attributes can be provided either at session or media level if multiple languages in the session description or media use multiple languages, in which case the order of the attributes indicates the order of importance of the various languages in the session or media from most important to least important.

In general, sending session descriptions consisting of multiple languages is discouraged. Instead, multiple descriptions SHOULD be sent describing the session, one in each language. However, this is not possible with all transport mechanisms, and so multiple sdplang attributes are allowed although NOT RECOMMENDED.

The "sdplang" attribute value must be a single RFC 3066 language tag in US-ASCII [9]. It is not dependent on the charset attribute. An "sdplang" attribute SHOULD be specified when a session is of sufficient scope to cross geographic boundaries where the language of recipients cannot be assumed, or where the session is in a different language from the locally assumed norm.

a=lang:<language tag>

This can be a session-level attribute or a media-level attribute. As a session-level attribute, it specifies the default language for the session being described. As a media-level attribute, it specifies the language for that media,



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 909
Keysight, EX-1045
Page 909

overriding any session-level language specified. Multiple lang attributes can be provided either at session or media level if the session description or media use multiple languages, in which case the order of the attributes indicates the order of importance of the various languages in the session or media from most important to least important.

The "lang" attribute value must be a single RFC 3066 language tag in US-ASCII [9]. It is not dependent on the charset attribute. A "lang" attribute SHOULD be specified when a session is of sufficient scope to cross geographic boundaries where the language of recipients cannot be assumed, or where the session is in a different language from the locally assumed norm.

a=framerate:<frame rate>

This gives the maximum video frame rate in frames/sec. It is intended as a recommendation for the encoding of video data. Decimal representations of fractional values using the notation "<integer>.<fraction>" are allowed. It is a media-level attribute, defined only for video media, and it is not dependent on charset.

a=quality:<quality>

This gives a suggestion for the quality of the encoding as an integer value. The intention of the quality attribute for video is to specify a non-default trade-off between frame-rate and still-image quality. For video, the value is in the range 0 to 10, with the following suggested meaning:

10 - the best still-image quality the compression scheme can give.
5 - the default behaviour given no quality suggestion.
0 - the worst still-image quality the codec designer thinks is still usable.

It is a media-level attribute, and it is not dependent on charset.

a=fmtp:<format> <format specific parameters>

This attribute allows parameters that are specific to a particular format to be conveyed in a way that SDP does not have to understand them. The format must be one of the formats specified for the media. Format-specific parameters may be any set of parameters required to be conveyed by SDP and given



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 910
Keysight, EX-1045
Page 910

unchanged to the media tool that will use this format. At most one instance of this attribute is allowed for each format.

It is a media-level attribute, and it is not dependent on charset.

## 7. Security Considerations

SDP is frequently used with the Session Initiation Protocol [15] using the offer/answer model [17] to agree on parameters for unicast sessions. When used in this manner, the security considerations of those protocols apply.

SDP is a session description format that describes multimedia sessions. Entities receiving and acting upon an SDP message SHOULD be aware that a session description cannot be trusted unless it has been obtained by an authenticated transport protocol from a known and trusted source. Many different transport protocols may be used to distribute session description, and the nature of the authentication will differ from transport to transport. For some transports, security features are often not deployed. In case a session description has not been obtained in a trusted manner, the endpoint SHOULD exercise care because, among other attacks, the media sessions received may not be the intended ones, the destination where media is sent to may not be the expected one, any of the parameters of the session may be incorrect, or the media security may be compromised. It is up to the endpoint to make a sensible decision taking into account the security risks of the application and the user preferences and may decide to ask the user whether or not to accept the session.

One transport that can be used to distribute session descriptions is the Session Announcement Protocol (SAP). SAP provides both encryption and authentication mechanisms, but due to the nature of session announcements it is likely that there are many occasions where the originator of a session announcement cannot be authenticated because the originator is previously unknown to the receiver of the announcement and because no common public key infrastructure is available.

On receiving a session description over an unauthenticated transport mechanism or from an untrusted party, software parsing the session should take a few precautions. Session descriptions contain information required to start software on the receiver's system. Software that parses a session description MUST NOT be able to start other software except that which is specifically configured as appropriate software to participate in multimedia sessions. It is normally considered inappropriate for software parsing a session



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 911
Keysight, EX-1045
Page 911

description to start, on a user's system, software that is appropriate to participate in multimedia sessions, without the user first being informed that such software will be started and giving the user's consent. Thus, a session description arriving by session announcement, email, session invitation, or WWW page MUST NOT deliver the user into an interactive multimedia session unless the user has explicitly pre-authorised such action. As it is not always simple to tell whether or not a session is interactive, applications that are unsure should assume sessions are interactive.

In this specification, there are no attributes that would allow the recipient of a session description to be informed to start multimedia tools in a mode where they default to transmitting. Under some circumstances it might be appropriate to define such attributes. If this is done, an application parsing a session description containing such attributes SHOULD either ignore them or inform the user that joining this session will result in the automatic transmission of multimedia data. The default behaviour for an unknown attribute is to ignore it.

In certain environments, it has become common for intermediary systems to intercept and analyse session descriptions contained within other signalling protocols. This is done for a range of purposes, including but not limited to opening holes in firewalls to allow media streams to pass, or to mark, prioritize, or block traffic selectively. In some cases, such intermediary systems may modify the session description, for example, to have the contents of the session description match NAT bindings dynamically created. These behaviours are NOT RECOMMENDED unless the session description is conveyed in such a manner that allows the intermediary system to conduct proper checks to establish the authenticity of the session description, and the authority of its source to establish such communication sessions. SDP by itself does not include sufficient information to enable these checks: they depend on the encapsulating protocol (e.g., SIP or RTSP).

Use of the "k=" field poses a significant security risk, since it conveys session encryption keys in the clear. SDP MUST NOT be used to convey key material, unless it can be guaranteed that the channel over which the SDP is delivered is both private and authenticated. Moreover, the "k=" line provides no way to indicate or negotiate cryptographic key algorithms. As it provides for only a single symmetric key, rather than separate keys for confidentiality and integrity, its utility is severely limited. The use of the "k=" line is NOT RECOMMENDED, as discussed in Section 5.12.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 912
Keysight, EX-1045
Page 912

## 8. IANA Considerations

### 8.1. The "application/sdp" Media Type

One media type registration from RFC 2327 is to be updated, as defined below.

To: ietf-types@iana.org
Subject: Registration of media type "application/sdp"

Type name: application

Subtype name: sdp

Required parameters: None.

Optional parameters: None.

Encoding considerations:
SDP files are primarily UTF-8 format text. The "a=charset:" attribute may be used to signal the presence of other character sets in certain parts of an SDP file (see Section 6 of RFC 4566). Arbitrary binary content cannot be directly represented in SDP.

Security considerations:
See Section 7 of RFC 4566

Interoperability considerations:
See RFC 4566

Published specification:
See RFC 4566

Applications which use this media type:
Voice over IP, video teleconferencing, streaming media, instant messaging, among others. See also Section 3 of RFC 4566.

Additional information:

Magic number(s): None.
File extension(s): The extension ".sdp" is commonly used.
Macintosh File Type Code(s): "sdp "

Person & email address to contact for further information:
Mark Handley <M.Handley@cs.ucl.ac.uk>
Colin Perkins <csp@csperkins.org>
IETF MMUSIC working group <mmusic@ietf.org>



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 913
Keysight, EX-1045
Page 913

Intended usage: COMMON

Author/Change controller:
Authors of RFC 4566
IETF MMUSIC working group delegated from the IESG

### 8.2. Registration of Parameters

There are seven field names that may be registered with IANA. Using the terminology in the SDP specification Backus-Naur Form (BNF), they are "media", "proto", "fmt", "att-field", "bwtype", "nettype", and "addrtype".

#### 8.2.1. Media Types ("media")

The set of media types is intended to be small and SHOULD NOT be extended except under rare circumstances. The same rules should apply for media names as for top-level media content types, and where possible the same name should be registered for SDP as for MIME. For media other than existing top-level media content types, a Standards Track RFC MUST be produced for a new top-level content type to be registered, and the registration MUST provide good justification why no existing media name is appropriate (the "Standards Action" policy of RFC 2434 [8].

This memo registers the media types "audio", "video", "text", "application", and "message".

Note: The media types "control" and "data" were listed as valid in the previous version of this specification [6]; however, their semantics were never fully specified and they are not widely used. These media types have been removed in this specification, although they still remain valid media type capabilities for a SIP user agent as defined in RFC 3840 [24]. If these media types are considered useful in the future, a Standards Track RFC MUST be produced to document their use. Until that is done, applications SHOULD NOT use these types and SHOULD NOT declare support for them in SIP capabilities declarations (even though they exist in the registry created by RFC 3840).

#### 8.2.2. Transport Protocols ("proto")

The "proto" field describes the transport protocol used. This SHOULD reference a standards-track protocol RFC. This memo registers three values: "RTP/AVP" is a reference to RTP [19] used under the RTP Profile for Audio and Video Conferences with Minimal Control [20]



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 914
Keysight, EX-1045
Page 914

running over UDP/IP, "RTP/SAVP" is a reference to the Secure Real-time Transport Protocol [23], and "udp" indicates an unspecified protocol over UDP.

If other RTP profiles are defined in the future, their "proto" name SHOULD be specified in the same manner. For example, an RTP profile whose short name is "XYZ" would be denoted by a "proto" field of "RTP/XYZ".

New transport protocols SHOULD be registered with IANA. Registrations MUST reference an RFC describing the protocol. Such an RFC MAY be Experimental or Informational, although it is preferable that it be Standards Track. Registrations MUST also define the rules by which their "fmt" namespace is managed (see below).

### 8.2.3. Media Formats ("fmt")

Each transport protocol, defined by the "proto" field, has an associated "fmt" namespace that describes the media formats that may be conveyed by that protocol. Formats cover all the possible encodings that might want to be transported in a multimedia session.

RTP payload formats under the "RTP/AVP" and "RTP/SAVP" profiles MUST use the payload type number as their "fmt" value. If the payload type number is dynamically assigned by this session description, an additional "rtpmap" attribute MUST be included to specify the format name and parameters as defined by the media type registration for the payload format. It is RECOMMENDED that other RTP profiles that are registered (in combination with RTP) as SDP transport protocols specify the same rules for the "fmt" namespace.

For the "udp" protocol, new formats SHOULD be registered. Use of an existing media subtype for the format is encouraged. If no media subtype exists, it is RECOMMENDED that a suitable one be registered through the IETF process [31] by production of, or reference to, a standards-track RFC that defines the transport protocol for the format.

For other protocols, formats MAY be registered according to the rules of the associated "proto" specification.

Registrations of new formats MUST specify which transport protocols they apply to.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 915
Keysight, EX-1045
Page 915

### 8.2.4. Attribute Names ("att-field")

Attribute field names ("att-field") MUST be registered with IANA and documented, because of noticeable issues due to conflicting attributes under the same name. Unknown attributes in SDP are simply ignored, but conflicting ones that fragment the protocol are a serious problem.

New attribute registrations are accepted according to the "Specification Required" policy of RFC 2434, provided that the specification includes the following information:

- contact name, email address, and telephone number
- attribute name (as it will appear in SDP)
- long-form attribute name in English
- type of attribute (session level, media level, or both)
- whether the attribute value is subject to the charset attribute
- a one-paragraph explanation of the purpose of the attribute
- a specification of appropriate attribute values for this attribute

The above is the minimum that IANA will accept. Attributes that are expected to see widespread use and interoperability SHOULD be documented with a standards-track RFC that specifies the attribute more precisely.

Submitters of registrations should ensure that the specification is in the spirit of SDP attributes, most notably that the attribute is platform independent in the sense that it makes no implicit assumptions about operating systems and does not name specific pieces of software in a manner that might inhibit interoperability.

IANA has registered the following initial set of attribute names ("att-field" values), with definitions as in Section 6 of this memo (these definitions update those in RFC 2327):



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 916
Keysight, EX-1045
Page 916

| Name | Session or Media level? | Dependent on charset? |
|---|---|---|
| cat | Session | No |
| keywds | Session | Yes |
| tool | Session | No |
| ptime | Media | No |
| maxptime | Media | No |
| rtpmap | Media | No |
| recvonly | Either | No |
| sendrecv | Either | No |
| sendonly | Either | No |
| inactive | Either | No |
| orient | Media | No |
| type | Session | No |
| charset | Session | No |
| sdplang | Either | No |
| lang | Either | No |
| framerate | Media | No |
| quality | Media | No |
| fmtp | Media | No |

#### 8.2.5. Bandwidth Specifiers ("bwtype")

A proliferation of bandwidth specifiers is strongly discouraged.

New bandwidth specifiers ("bwtype" fields) MUST be registered with IANA. The submission MUST reference a standards-track RFC specifying the semantics of the bandwidth specifier precisely, and indicating when it should be used, and why the existing registered bandwidth specifiers do not suffice.

IANA has registered the bandwidth specifiers "CT" and "AS" with definitions as in Section 5.8 of this memo (these definitions update those in RFC 2327).

#### 8.2.6. Network Types ("nettype")

New network types (the "nettype" field) may be registered with IANA if SDP needs to be used in the context of non-Internet environments. Although these are not normally the preserve of IANA, there may be circumstances when an Internet application needs to interoperate with a non-Internet application, such as when gatewaying an Internet telephone call into the Public Switched Telephone Network (PSTN). The number of network types should be small and should be rarely extended. A new network type cannot be registered without registering at least one address type to be used with that network



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 917
Keysight, EX-1045
Page 917

type. A new network type registration MUST reference an RFC that gives details of the network type and address type and specifies how and when they would be used.

IANA has registered the network type "IN" to represent the Internet, with definition as in Sections 5.2 and 5.7 of this memo (these definitions update those in RFC 2327).

### 8.2.7. Address Types ("addrtype")

New address types ("addrtype") may be registered with IANA. An address type is only meaningful in the context of a network type, and any registration of an address type MUST specify a registered network type or be submitted along with a network type registration. A new address type registration MUST reference an RFC giving details of the syntax of the address type. Address types are not expected to be registered frequently.

IANA has registered the address types "IP4" and "IP6" with definitions as in Sections 5.2 and 5.7 of this memo (these definitions update those in RFC 2327).

### 8.2.8. Registration Procedure

In the RFC documentation that registers SDP "media", "proto", "fmt", "bwtype", "nettype", and "addrtype" fields, the authors MUST include the following information for IANA to place in the appropriate registry:

- contact name, email address, and telephone number
- name being registered (as it will appear in SDP)
- long-form name in English
- type of name ("media", "proto", "fmt", "bwtype", "nettype", or "addrtype")
- a one-paragraph explanation of the purpose of the registered name
- a reference to the specification for the registered name (this will typically be an RFC number)

IANA may refer any registration to the IESG for review, and may request revisions to be made before a registration will be made.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 918
Keysight, EX-1045
Page 918

### 8.3. Encryption Key Access Methods

The IANA previously maintained a table of SDP encryption key access method ("enckey") names. This table is obsolete, since the "k=" line is not extensible. New registrations MUST NOT be accepted.

## 9. SDP Grammar

This section provides an Augmented BNF grammar for SDP. ABNF is defined in [4].

```
; SDP Syntax
session-description = proto-version
                      origin-field
                      session-name-field
                      information-field
                      uri-field
                      email-fields
                      phone-fields
                      connection-field
                      bandwidth-fields
                      time-fields
                      key-field
                      attribute-fields
                      media-descriptions

proto-version =       %x76 "=" 1*DIGIT CRLF
                      ;this memo describes version 0

origin-field =        %x6f "=" username SP sess-id SP sess-version SP
                      nettype SP addrtype SP unicast-address CRLF

session-name-field =  %x73 "=" text CRLF

information-field =   [%x69 "=" text CRLF]

uri-field =           [%x75 "=" uri CRLF]

email-fields =        *(%x65 "=" email-address CRLF)

phone-fields =        *(%x70 "=" phone-number CRLF)

connection-field =    [%x63 "=" nettype SP addrtype SP
                      connection-address CRLF]
                      ;a connection field must be present
                      ;in every media description or at the
                      ;session-level
```



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 919
Keysight, EX-1045
Page 919

```
   bandwidth-fields =    *(%x62 "=" bwtype ":" bandwidth CRLF)

   time-fields =         1*( %x74 "=" start-time SP stop-time
                         *(CRLF repeat-fields) CRLF)
                         [zone-adjustments CRLF]

   repeat-fields =       %x72 "=" repeat-interval SP typed-time
                         1*(SP typed-time)

   zone-adjustments =    %x7a "=" time SP ["-"] typed-time
                         *(SP time SP ["-"] typed-time)

   key-field =           [%x6b "=" key-type CRLF]

   attribute-fields =    *(%x61 "=" attribute CRLF)

   media-descriptions =  *( media-field
                         information-field
                         *connection-field
                         bandwidth-fields
                         key-field
                         attribute-fields )

   media-field =         %x6d "=" media SP port ["/" integer]
                         SP proto 1*(SP fmt) CRLF

   ; sub-rules of 'o='
   username =            non-ws-string
                         ;pretty wide definition, but doesn't
                         ;include space

   sess-id =             1*DIGIT
                         ;should be unique for this username/host

   sess-version =        1*DIGIT

   nettype =             token
                         ;typically "IN"

   addrtype =            token
                         ;typically "IP4" or "IP6"

   ; sub-rules of 'u='
   uri =                 URI-reference
                         ; see RFC 3986
```



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 920
Keysight, EX-1045
Page 920

```
   ; sub-rules of 'e=', see RFC 2822 for definitions
   email-address        = address-and-comment / dispname-and-address
                          / addr-spec
   address-and-comment  = addr-spec 1*SP "(" 1*email-safe ")"
   dispname-and-address = 1*email-safe 1*SP "<" addr-spec ">"

   ; sub-rules of 'p='
   phone-number =        phone *SP "(" 1*email-safe ")" /
                         1*email-safe "<" phone ">" /
                         phone

   phone =               ["+"] DIGIT 1*(SP / "-" / DIGIT)

   ; sub-rules of 'c='
   connection-address =  multicast-address / unicast-address

   ; sub-rules of 'b='
   bwtype =              token

   bandwidth =           1*DIGIT

   ; sub-rules of 't='
   start-time =          time / "0"

   stop-time =           time / "0"

   time =                POS-DIGIT 9*DIGIT
                         ; Decimal representation of NTP time in
                         ; seconds since 1900.  The representation
                         ; of NTP time is an unbounded length field
                         ; containing at least 10 digits.  Unlike the
                         ; 64-bit representation used elsewhere, time
                         ; in SDP does not wrap in the year 2036.

   ; sub-rules of 'r=' and 'z='
   repeat-interval =     POS-DIGIT *DIGIT [fixed-len-time-unit]

   typed-time =          1*DIGIT [fixed-len-time-unit]

   fixed-len-time-unit = %x64 / %x68 / %x6d / %x73

   ; sub-rules of 'k='
   key-type =            %x70 %x72 %x6f %x6d %x70 %x74 /     ; "prompt"
                         %x63 %x6c %x65 %x61 %x72 ":" text / ; "clear:"
                         %x62 %x61 %x73 %x65 "64:" base64 /  ; "base64:"
                         %x75 %x72 %x69 ":" uri              ; "uri:"

   base64      =         *base64-unit [base64-pad]
```



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 921
Keysight, EX-1045
Page 921

```
   base64-unit =          4base64-char
   base64-pad  =          2base64-char "==" / 3base64-char "="
   base64-char =          ALPHA / DIGIT / "+" / "/"

   ; sub-rules of 'a='
   attribute =            (att-field ":" att-value) / att-field

   att-field =            token

   att-value =            byte-string

   ; sub-rules of 'm='
   media =                token
                          ;typically "audio", "video", "text", or
                          ;"application"

   fmt =                  token
                          ;typically an RTP payload type for audio
                          ;and video media

   proto  =               token *("/" token)
                          ;typically "RTP/AVP" or "udp"

   port =                 1*DIGIT

   ; generic sub-rules: addressing
   unicast-address =      IP4-address / IP6-address / FQDN / extn-addr

   multicast-address =    IP4-multicast / IP6-multicast / FQDN
                          / extn-addr

   IP4-multicast =        m1 3( "." decimal-uchar )
                          "/" ttl [ "/" integer ]
                          ; IPv4 multicast addresses may be in the
                          ; range 224.0.0.0 to 239.255.255.255

   m1 =                   ("22" ("4"/"5"/"6"/"7"/"8"/"9")) /
                          ("23" DIGIT )

   IP6-multicast =        hexpart [ "/" integer ]
                          ; IPv6 address starting with FF

   ttl =                  (POS-DIGIT *2DIGIT) / "0"

   FQDN =                 4*(alpha-numeric / "-" / ".")
                          ; fully qualified domain name as specified
                          ; in RFC 1035 (and updates)
```



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 922
Keysight, EX-1045
Page 922

```
   IP4-address =         b1 3("." decimal-uchar)

   b1 =                  decimal-uchar
                         ; less than "224"

   ; The following is consistent with RFC 2373 [30], Appendix B.
   IP6-address =         hexpart [ ":" IP4-address ]

   hexpart =             hexseq / hexseq "::" [ hexseq ] /
                         "::" [ hexseq ]

   hexseq  =             hex4 *( ":" hex4)

   hex4    =             1*4HEXDIG

   ; Generic for other address families
   extn-addr =           non-ws-string

   ; generic sub-rules: datatypes
   text =                byte-string
                         ;default is to interpret this as UTF8 text.
                         ;ISO 8859-1 requires "a=charset:ISO-8859-1"
                         ;session-level attribute to be used

   byte-string =         1*(%x01-09/%x0B-0C/%x0E-FF)
                         ;any byte except NUL, CR, or LF

   non-ws-string =       1*(VCHAR/%x80-FF)
                         ;string of visible characters

   token-char =          %x21 / %x23-27 / %x2A-2B / %x2D-2E / %x30-39
                         / %x41-5A / %x5E-7E

   token =               1*(token-char)

   email-safe =          %x01-09/%x0B-0C/%x0E-27/%x2A-3B/%x3D/%x3F-FF
                         ;any byte except NUL, CR, LF, or the quoting
                         ;characters ()<>

   integer =             POS-DIGIT *DIGIT

   ; generic sub-rules: primitives
   alpha-numeric =       ALPHA / DIGIT

   POS-DIGIT =           %x31-39 ; 1 - 9
```



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 923
Keysight, EX-1045
Page 923

```
   decimal-uchar =       DIGIT
                         / POS-DIGIT DIGIT
                         / ("1" 2*(DIGIT))
                         / ("2" ("0"/"1"/"2"/"3"/"4") DIGIT)
                         / ("2" "5" ("0"/"1"/"2"/"3"/"4"/"5"))

   ; external references:
         ; ALPHA, DIGIT, CRLF, SP, VCHAR: from RFC 4234
         ; URI-reference: from RFC 3986
         ; addr-spec: from RFC 2822
```

## 10. Summary of Changes from RFC 2327

The memo has been significantly restructured, incorporating a large number of clarifications to the specification in light of use. With the exception of those items noted below, the changes to the memo are intended to be backward-compatible clarifications. However, due to inconsistencies and unclear definitions in RFC 2327 it is likely that some implementations interpreted that memo in ways that differ from this version of SDP.

The ABNF grammar in Section 9 has been extensively revised and updated, correcting a number of mistakes and incorporating the RFC 3266 IPv6 extensions. Known inconsistencies between the grammar and the specification text have been resolved.

A media type registration for SDP is included. Requirements for the registration of attributes and other parameters with IANA have been clarified and tightened (Section 8). It is noted that "text" and "message" are valid media types for use with SDP, but that "control" and "data" are under-specified and deprecated.

RFC 2119 terms are now used throughout to specify requirements levels. Certain of those requirements, in particular in relation to parameter registration, are stricter than those in RFC 2327.

The "RTP/SAVP" RTP profile and its "fmt" namespace are registered.

The attributes "a=inactive" and "a=maxptime" have been added.

RFC 2327 mandated that either "e=" or "p=" was required. Both are now optional, to reflect actual usage.

The significant limitations of the "k=" field are noted, and its use is deprecated.

Most uses of the "x-" prefix notation for experimental parameters are disallowed and the other uses are deprecated.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 924
Keysight, EX-1045
Page 924

## 11. Acknowledgements

Many people in the IETF Multiparty Multimedia Session Control (MMUSIC) working group have made comments and suggestions contributing to this document. In particular, we would like to thank Eve Schooler, Steve Casner, Bill Fenner, Allison Mankin, Ross Finlayson, Peter Parnes, Joerg Ott, Carsten Bormann, Steve Hanna, Jonathan Lennox, Keith Drage, Sean Olson, Bernie Hoeneisen, Jonathan Rosenberg, John Elwell, Flemming Andreasen, Jon Peterson, and Spencer Dawkins.

## 12. References

### 12.1. Normative References

[1] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[2] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[3] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[4] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005.

[5] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[6] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

[7] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[8] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[9] Alvestrand, H., "Tags for the Identification of Languages", BCP 47, RFC 3066, January 2001.

[10] Olson, S., Camarillo, G., and A. Roach, "Support for IPv6 in Session Description Protocol (SDP)", RFC 3266, June 2002.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 925
Keysight, EX-1045
Page 925

[11] Faltstrom, P., Hoffman, P., and A. Costello, "Internationalizing Domain Names in Applications (IDNA)", RFC 3490, March 2003.

[12] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 3548, July 2003.

### 12.2. Informative References

[13] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation", RFC 1305, March 1992.

[14] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", RFC 2974, October 2000.

[15] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[16] Schulzrinne, H., Rao, A., and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

[17] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[18] Camarillo, G., Eriksson, G., Holler, J., and H. Schulzrinne, "Grouping of Media Lines in the Session Description Protocol (SDP)", RFC 3388, December 2002.

[19] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[20] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, July 2003.

[21] Casner, S., "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", RFC 3556, July 2003.

[22] Huitema, C., "Real Time Control Protocol (RTCP) attribute in Session Description Protocol (SDP)", RFC 3605, October 2003.

[23] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 926
Keysight, EX-1045
Page 926

[24] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)", RFC 3840, August 2004.

[25] Westerlund, M., "A Transport Independent Bandwidth Modifier for the Session Description Protocol (SDP)", RFC 3890, September 2004.

[26] International Telecommunication Union, "H.323 extended for loosely coupled conferences", ITU Recommendation H.332, September 1998.

[27] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K. Norrman, "Key Management Extensions for Session Description Protocol (SDP) and Real Time Streaming Protocol (RTSP)", RFC 4567, July 2006.

[28] Andreasen, F., Baugher, M., and D. Wing, "Session Description Protocol (SDP) Security Descriptions for Media Streams", RFC 4568, July 2006.

[29] Resnick, P., "Internet Message Format", RFC 2822, April 2001.

[30] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 2373, July 1998.

[31] Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", BCP 13, RFC 4288, December 2005.



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 927
Keysight, EX-1045
Page 927

## Authors' Addresses

Mark Handley
University College London
Department of Computer Science
Gower Street
London WC1E 6BT
UK

EMail: M.Handley@cs.ucl.ac.uk

Van Jacobson
Packet Design
2465 Latham Street
Mountain View, CA 94040
USA

EMail: van@packetdesign.com

Colin Perkins
University of Glasgow
Department of Computing Science
17 Lilybank Gardens
Glasgow G12 8QQ
UK

EMail: csp@csperkins.org



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 928
Keysight, EX-1045
Page 928

Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).



https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 929
Keysight, EX-1045
Page 929

https://web.archive.org/web/20070128163222/http://tools.ietf.org:80/html/rfc4566

Login

# PhoneBoy's Security Theater

## Views on Information Security from the guy who brought you ye olde PhoneBoy FireWall-1 FAQ

## Blast from the CHKP Past: Can't Talk to Translated IP from Internal Net

It's pretty obvious from looking through the number of 404s I'm seeing in Google's Webmaster tools that a lot of pages still link to old stuff I wrote about Check Point FireWall-1. I'm actually trying to "fix" these 404s now by resurrecting some of the old content. Not updating it, of course, but at least making the links point to something semi-useful, if historical.

This is one of those articles, obviously not at it's original URL, but the original URLs will point here. What amazes me about this particular article is that it's still relevant today as NAT really hasn't fundamentally changed in the Check Point products for some time. The basic concepts are still the same, too, and other than the implementation details, is probably relevant for other security products, too.

Bottom line: NAT only works if the firewall is in the path of the communication. How do you know? Follow the bouncing packet, otherwise known as Troubleshooting 101.

Hit the break to see this old FAQ in all its ASCII network mapped glory.

Read more »

Upvote 0

0 responses // Posted about 1 month ago

## The TSA and A Lesson in Data Loss Prevention

From PhoneBoy Speaks Ep 305: Threat? What Threat?

> In sealed court documents accidentally leaked on a US Federal government website, the US government basically admits that there has been no attempted domestic hijackings of any kind in the 12 years since 9/11. Furthermore, at least as of mid-2011, terrorist threat groups present in the Homeland are not known to be actively plotting against civil aviation targets or airports.

It gets better. Jonathan Corbett, the guy who has been challenging the feds on the constitutionality and the effectiveness of the TSA in the courts, was told by the US Justice Department to take down his comments about a sealed document he wasn't allowed to talk about, but the government themselves



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 931
Keysight, EX-1045
Page 931

posted the unredacted version on their site and other sites made it public.

The same is true for Corbett's comments about those documents, the Justice Department asked him to remove from his site, but they apparently forgot to tell Google as it was still in their cache!

This is a clear demonstration of what can happen if confidential data inadvertently leaves your organization. Clearly this unredacted, sealed legal brief should have never been made public. Thankfully, in this case, it was, but surely some folks in the US Government weren't happy with this lapse. Neither would your employer if it's your company's secret plans for world domination that got leaked.

Likewise, if you inadvertently leak information on your own website, or someone else does, even if you can manage to get it taken down, Google takes a while to forget it saw it. The rest of the Internet won't necessarily forget, though. Ever.

So what are your plans for Data Loss Prevention in your organization? Are you even employing any DLP technology? Is it actually catching real incidents of data loss or is everyone going around it?

I'm not saying DLP is a panacea or 100% effective, but if you're not doing it, then you don't have any idea where your corporate information might end up.

Upvote 0

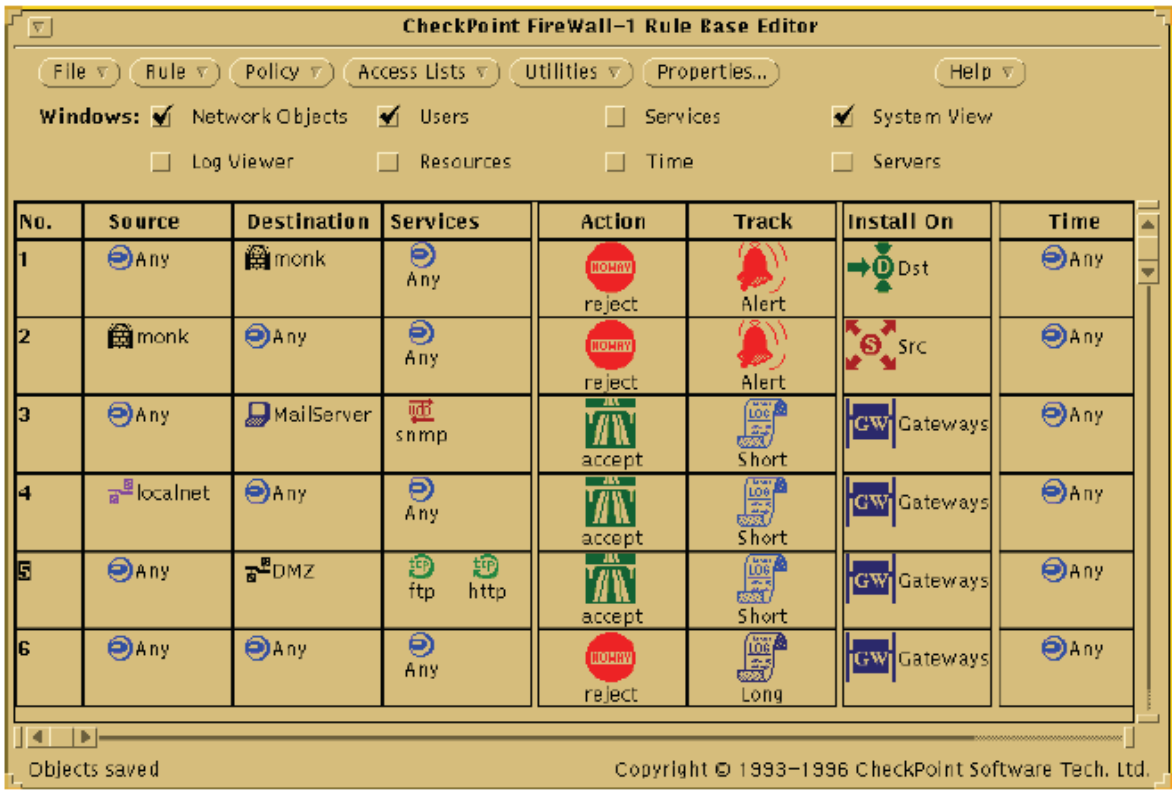
0 responses // Posted 3 months ago

## FireWall-1 User Interface from 3.0

If you ran Check Point FireWall-1 on SunOS or Solaris, you probably remember this screen:

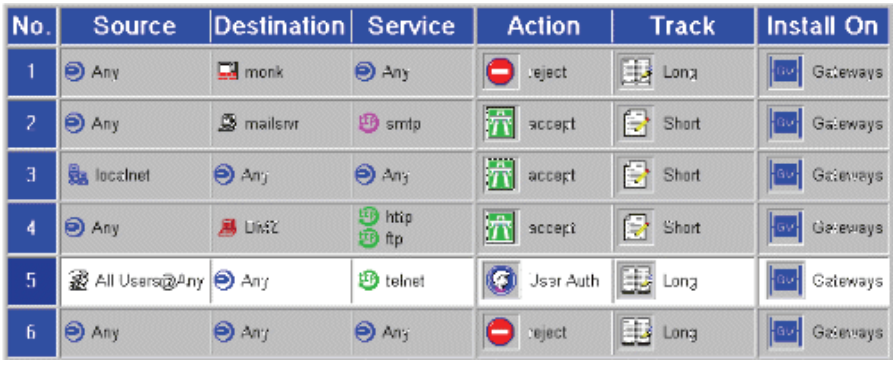


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 932
Keysight, EX-1045
Page 932




This is a screen capture from fwui, the management interface for FireWall-1. Actually FireWall-1 2.0 and 2.1 looked like this too.

Meanwhile if you ran the management interface on Windows, you were presented with something like this:




You could actually run the firewall on Windows NT as of 2.1, though it took until 2.1c until it was reasonably stable.

Actually you could run something that looked like the Windows UI on Solaris with Motif, which Check Point charged extra for. Eventually the Motif and the OpenLook GUI were deprecated and what we now refer to as SmartDashboad only runs on Windows.

https://web.archive.org/web/20140102143305/http://phoneboy.net/

And, of course, there are far more GUI clients these days--about a dozen or so if I remember correctly. But then again, the product does so much more today than it did back in the mid 1990s.

And yes, I remember those days quiet fondly.
Upvote 0

2 responses // Posted 5 months ago

## Blast from the CHKP Past: Can't Get Putkeys to Work

For those of you who never had to work with Check Point FireWall-1 4.1 and earlier, you're lucky you likely never EVER had to use putkeys.

The fw putkey was used to establish authentication between the Management and Firewall modules. The problem was: the authentication would frequently break. Especially in larger, distributed environments. Often for reasons that no one outside of a few developers in Tel Aviv never fully understood.

The eventual replacement for fw putkey was SIC, which was added in FireWall-1 NG (5.0). It is based on certificates and is way easier to set up. It's also far less prone to random breakage.

The following classic article is presented for nostalgia purposes only. Hopefully no one in their right mind still needs this article, which was very popular on my FireWall-1 FAQ back in the day. It was the collective wisdom of my peers and my own experience as of around 2000 or so.

Read more »

Upvote 1

2 responses // Posted 5 months ago

## Moving The Security Theater Elsewhere

I decided to move PhoneBoy's Security Theater off of Wordpress onto Posthaven.

On the plus side, it's one less instance of Wordpress to maintain. On the minus side, I had to copy/paste the old articles in since there is no way to import Wordpress as of yet. But it was under 100 articles and it was kind of nice to see my thoughts on security from the last several years again.

The reality is, the threats haven't changed. We've had to evolve the tools in order to address increasingly complex threats. You can look at Check Point's own product evolution to get a sense of that.

Also, the basics haven't changed. A lot of security risks can be sufficiently mitigated by properly segmenting your networks, using sufficiently complex passwords that are unique, proper monitoring of the logs of your security devices, and keeping your software up-to-date with the latest security patches.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 934
Keysight, EX-1045
Page 934

**Tags**

- administrivia

Upvote 0

0 responses // Posted 5 months ago

## Fun with Compliance

Earlier this week, I hung out with Jeremy Kaye, one of our in-house compliance experts at Check Point:

http://www.youtube.com/embed/uvL6HdlrW08

While I've been doing InfoSec for a while, or at least working in companies that sell InfoSec products, compliance isn't something I've had a ton of direct experience with. Sure, Check Point customers used our products to help meet various compliance regulations, but until Check Point acquired DynaSec in 2011, there wasn't a team inside Check Point dedicated to this topic.

While we had some technical challenges with the Google+ Hangout itself (and it was the first one we did at Check Point), I think the conversation with Jeremy went fairly well. The questions I asked where ones I've always wanted answers to. Like, what good is compliance? Why does it seem like compliance is in the eye of the auditor? Why so many regulations anyway?

The big takeaway for me from this conversation is that security should drive your compliance efforts, not the other way around. Because chances are, if you have a strong information security program in place already, compliance is pretty straightforward, no matter which regulations you have to comply with.

**Tags**

- compliance

Upvote 0

0 responses // Posted 6 months ago

## Check Point 600 Appliance: Big Security for Small Business

Ok, this is a completely shameless plug for my employer. But it's really big. And really small at the same time. And my take on it, which wasn't cleared with the marketing folks, and thus my, albeit biased, opinion.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 935
Keysight, EX-1045
Page 935

The Check Point 600 Appliance, which was announced today at Interop, represents Check Point's refreshed entry into the SMB Security space. It provides the same security functionality you'd find in Check Point's larger appliances in something that fits into an SMB--both in terms of form factor and price. This includes Check Point's award-winning IPS, App Control, URL Filtering, Anti-Virus, Anti-Spam, VPN, oh and don't forget the firewall :)

If you're familiar with the SG80, which Check Point launched a couple years back, the new 600 Appliance looks a bit like that, though the internals are slightly different from the SG80. There are standard USB ports, Express Card and SD-card slots in the 600 as well as optional WiFi and ADSL ports. It also includes a revamped Web Interface that incorporates functionality from the UTM-1 EDGE and Safe@ appliances allowing full management of the security policy across all Software Blades.

Under the hood? It's nearly the same code that runs in the larger Check Point appliances--Check Point R75.20 running Embedded Gaia, to be exact. When you SSH or serial console into the appliance, you are presented with clish, which functions similar to how it does on one of the larger appliances. You can also drop into Expert mode for more advanced debugging, which again, works very similar to how its done on the larger gateways.

The main differences between the 600 and the Check Point 1100 Appliance, which was announced a few weeks ago are:

- Lower price: List price of a 600 is roughly $200 cheaper than the comparable 1100 model.
- Chassis color: Bright orange, like the old Safe@ boxes.
- Central Management: While the 1100 can be centrally managed with standard R75.46 or R76 management (standalone or Provider-1), the 600 can only be centrally managed by Check Point Cloud-Managed Security service.

In any case, I am truly excited about this as finally, SMBs can finally get the same Enterprise-grade security that the Fortune 100 relies on for a fraction of the cost--starting at $399.

Check Point's SMB Portal has information about the new appliances as well as how to acquire them.

**Tags**

- check point
- smb

Upvote 0

0 responses // Posted 8 months ago

## Check Point 2013 Security Report

When I was in Israel at the end of 2012, I was talking with the folks putting the finishing touches on the Check Point 2013 Security Report. Of course, since then, the report has been formally released and you can now read it for yourself. Here's a video preview of what you'll find in it:




Some of the data gathered for this report was related to the 3D Security Reports Check Point generated for customers during 2012 where we took a Security Gateway and either ran it in-line (in bridge mode) or plugged it into a mirror port on a customer's switch. It's worth pointing out that, in many cases, *a competitive security solution was already in place* and the Check Point Security Gateways were seeing stuff the other solutions were missing.

Other data for this report also came from SensorNet, ThreatCloud, and results from our Endpoint Security Best Practices Report, which is a great way to find out if your Windows PC is configured according to our Best Practices.

The most surprising statistics?

- 63% of the organizations surveyed had at least one malicious bot in their network.
- 43% of the organizations surveyed had traffic to/from an anonymizer service.

Of course, if you're knee deep in the security space, 0% of this is news to you.

**Tags**

- check point

Upvote 0

0 responses // Posted 9 months ago

## How to Not Be Like Burger King. Or Jeep.

On today's episode of PhoneBoy Speaks, I discuss how to prevent your Twitter account from being hacked like Burger King's account was. And today (after I recorded this episode), Jeep's Twitter account was also hacked. Of course, I can only do so much in a 5 minute podcast, and the topic itself of choosing strong passwords--and getting users to actually do it--has been covered ad-infinitum elsewhere.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 937
Keysight, EX-1045
Page 937

The fact is, passwords are not very secure. To be secure, they must be both long (number of characters) and high-entropy (more random, the better). Humans, as a lot, are not able to remember passwords that meet both of these requirements, so they cheat. They either write the passwords down, they use password management tools like LastPass or 1Password, or they just choose stupid passwords--usually the latter.

The best compromise I've seen is actually the Password Haystacks method that Steve Gibson came up with. All other things being equal, as long as you use all 4 different types of characters in your password, length wins. Because when it comes to guessing passwords, there is no such thing as "close."

Of course, if the password itself can't be guessed, surely you can compromise the password reset process, as was done with Mat Honan's widely publicized pwnage. Hopefully we can strengthen that too, but companies--especially ones that cater to non technical people--rarely err on the side of secure.

**Tags**

- social networking
- passwords

Upvote 0

0 responses // Posted 11 months ago

## Securing Data in the Cloud

On my podcast PhoneBoy Speaks today, I discussed (very briefly) the idea of doing information security in the cloud. Surely, I could talk and write volumes on the subject. I've even given presentations on the subject.

The reality is, virtualization changes the game in so many ways that it's hard to know where to begin. That said, my view starts with the most basic question: what is it we're ultimately trying to protect?

The good news is that the answer is still the same, regardless of whether physical servers on your premises are involved or some cloud services provider is: it's the data. Your job in information security is to ensure the Confidentiality, Integrity, and Availability of data to prevent Disclosure, Alteration, or Destruction of said data.

The bad news: the cloud makes this job a lot harder. The reality of bring your own device (BYOD) also makes this harder for much the same reason--less opportunities to inject the necessary controls to ensure data doesn't go where it's not supposed to.

Of course, it's not just about protecting the data. That part is actually pretty easy. Protecting in a way that allows it to be used in a convenient way, now that's a lot harder.

**Tags**

- cloud
- byod

Upvote 0

0 responses // Posted 11 months ago


- 1
- 2
- 3
- 4
- 5
- …
- Next ›
- Last »


- About PhoneBoy
- PhoneBoy on app.net
- PhoneBoy on Twitter
- PhoneBoy on Google+
- PhoneBoy on LinkedIn

## Dameon Welch-Abernathy



Parent, Writer, Podcaster, Coffee Achiever, VoIP/Telco Curious, C-List Information Security Celebrity, and Mobile Phone Connoisseur residing in Gig Harbor, WA, USA.

You might know him better as PhoneBoy.

**Subscribe by email »**
We'll email you when there are new posts here.

## Tags

- check point 26
- nokia 9
- wifi 6
- mobile phones 6



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 939
**Keysight, EX-1045**
**Page 939**

- iskoot 5
- social networking 5
- byod 5
- skype 4
- facebook 4
- voip 3
- software 3
- ios 3
- cpug 2
- CISSP 2
- firesheep 2
- iphone 2
- windows 2
- sourcefire 2
- ips 2
- cpshared 2
- compliance 2
- SecureClient 2
- jailbreak 2
- connectivity 2
- ve 1
- mobile 1
- sip 1
- ipv6 1
- telecom 1
- VPN 1
- securid 1
- See all 66 tags »

## Sites I Like

- The PhoneBoy Blog
- PhoneBoy's Personal Blog



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 940
**Keysight, EX-1045**
**Page 940**

https://web.archive.org/web/20120316165725/http://www.ndm.net:80/sourcefirestore/downloads/Sourcefire%20SSL%20Appliance%20Fact%20Sheet%20%28English%29.pdf

# Sourcefire SSL Appliance

## Strengthen Security with Secure Sockets Layer (SSL) Inspection



***The Sourcefire SSL Appliance decrypts SSL traffic at 1Gbps line rate to enable existing security appliances to effectively inspect SSL traffic. The SSL Appliance operates transparently on the network and supports both passive and inline network configurations.***

**Surveys show 25-35% of enterprise traffic is SSL-encrypted, and this number is up to 70% for select verticals.**

**SSL-encrypted communications are an easy vehicle for the following cybersecurity attacks:**

- Inbound attacks
- Spam
- Spyware and malware
- Viruses and worms
- Phishing
- Identity theft
- Information leaks

### SSL-ENCRYPTED TRAFFIC—AN EASY VEHICLE FOR CYBERSECURITY ATTACKS

SSL-encrypted traffic is exploding due to the enterprise-wide usage of cloud computing, secure e-commerce, Web 2.0 applications, email, and VPNs. Surveys show 25-35% of enterprise traffic is SSL-encrypted, and this number is up to 70% for select verticals. If not managed properly, SSL can leave a hole in any enterprise security architecture. Existing approaches to SSL-encrypted traffic range from passing everything to blocking everything. In some cases, companies deploy host-based Intrusion Prevention Systems (IPS) or install proxy SSL solutions, which can effectively inspect SSL but suffer from bottleneck issues and reduced network performance.

### SOURCEFIRE SSL APPLIANCE

#### Decrypts SSL Traffic at 1Gbps Line Rate

The Sourcefire SSL Appliance decrypts SSL traffic and sends it to existing security and network appliances via dedicated gigabit Ethernet links. This enables existing IPS appliances to identify risks normally hidden by SSL such as regulatory compliance violations, viruses, malware, data loss, and intrusion attempts. Once the SSL traffic has been inspected and approved, the SSL Appliance places the SSL-encrypted traffic back on the network for its final destination—all with minimal latency and without altering SSL packets. Unlike on-box SSL decryption solutions that use shared hardware resources for SSL decryption and IPS inspection, the Sourcefire architecture permits the SSL and IPS processes to run on separate systems, offloading all decryption and encryption requirements from the IPS. This provides users with greater IPS performance and scalability.

#### Operates Transparently on Network

The SSL Appliance is deployed as a transparent proxy and detects SSL sessions on all ports, not just the traditional port 443. It can run as a "bump-in-the-wire" and does not require network configuration, IP addressing or topology changes, or modification to client IP and web browser configurations. Further, transparent SSL proxies see all network traffic, not just SSL, and have the ability to cut-through non-SSL flows. Traditional SSL proxies, by comparison, require IP address configuration and possibly network topology changes to inspect traffic. These SSL proxies are now vulnerable to all attacks just as any other host or network element would be. Also, these proxies often assume that all SSL traffic is directed to port 443 and ignore SSL traffic on other ports.

**Security Functions:**

- Encryption: TLS 1.0, TLS 1.1, SSL3, partial SSL2
- Proxy Mode: Transparent
- Public Key Algorithms: RSA, DSA, DH
- Symmetric Key Algorithms: AES, 3DES, DES, RC4
- Hashing Algorithms: MD5, SHA-1
- RSA Keys: 512, 1024, 2048, 4096, 8172 bits

## Supports Passive and Inline Configurations

The SSL Appliance supports both passive and inline configurations. When deployed passively, it sends traffic to a Sourcefire IPS™ also running in passive mode. Passive deployment is most useful for gaining full visibility into network traffic and what vulnerabilities may be exploited. The SSL Appliance can also be deployed inline as a "bump-in-the-wire" and operate with an IPS running in either passive or inline mode. When both the SSL Appliance and the IPS are deployed inline, they can block malicious exploit traffic. All Sourcefire SSL Appliances ship with fail-open 4-port 1G copper or fiber interfaces. The SSL Appliance is versatile enough to inspect SSL traffic in both inbound and outbound configurations. With inbound SSL inspection, the appliance inspects traffic destined for an enterprise's web servers hosting SSL applications. With outbound SSL inspection, the appliance inspects SSL application traffic destined outside of the enterprise, such as Google Gmail traffic.

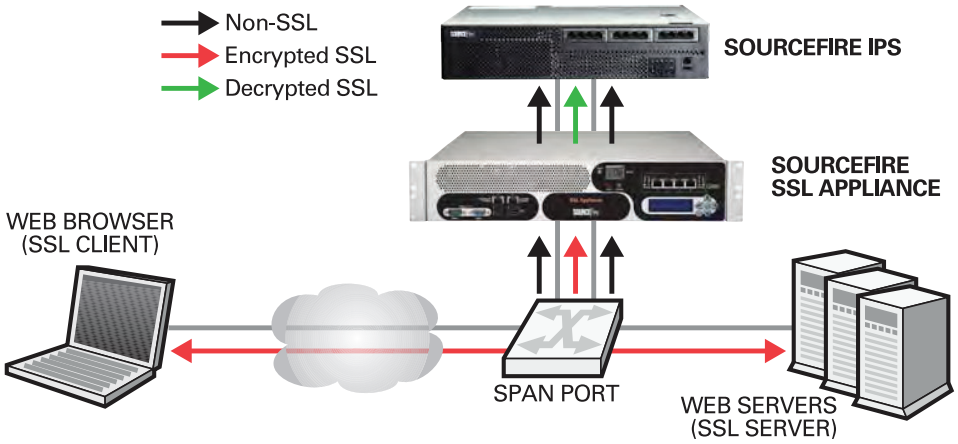



**Figure 1.** Passive IDS Configuration

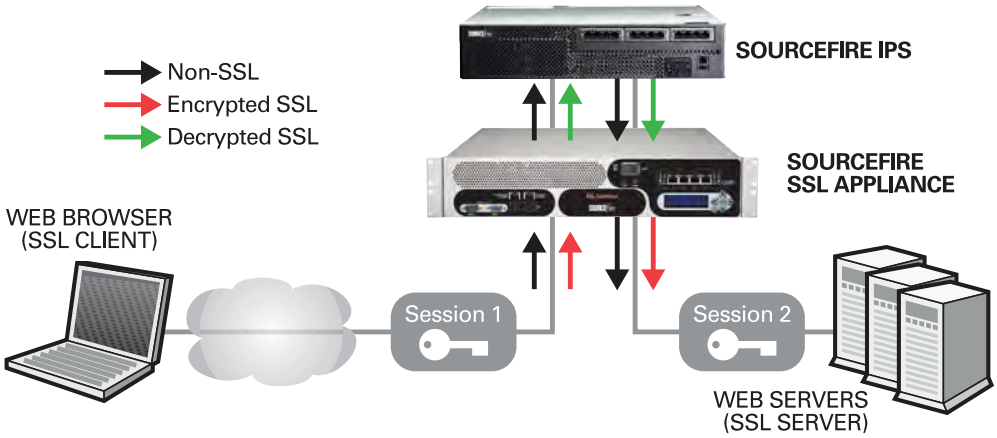



**Figure 2.** Inline IPS Configuration



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 943
Keysight, EX-1045
Page 943

## UNIQUE CAPABILITIES

The unique capabilities of the Sourcefire SSL Appliance remove risks arising from lack of visibility into SSL traffic while also maintaining the performance of security and network appliances.

- **Scalable Flow-based Processing:** At up to 1Gbps, the Sourcefire SSL Appliance supports the analysis of over 1,000,000 simultaneous flows.
- **High Connection Rate/Flow Count:** The Sourcefire SSL Appliance supports 50,000 concurrent SSL sessions. The setup and teardown rate of 2,900 SSL sessions per second is 10x higher than other solutions.
- **Line-rate Network Performance:**
  - » Non-SSL flows can be sent to the adjacent appliance or cut-through in less than 40 microseconds, minimizing delay for applications such as VoIP.
  - » Supports proxying for up to 1Gbps of SSL traffic for a variety of SSL versions and cipher suites.
- **Network Transparency:** The Sourcefire SSL Appliance can be deployed transparently to both end systems and intermediate networking elements and does not require network configuration, IP addressing or topology changes, or modification to client IP and web browser configurations.
- **Application Preservation:** Intercepted plaintext is delivered to security appliances as a regenerated TCP stream with the packet headers as they were received. This enables applications and appliances, such as Intrusion Detection System (IDS), IPS, Unified Threat Management (UTM), and Data Loss Prevention (DLP), to expand their scope to provide benefits for SSL-encrypted traffic.
- **Flexibility:**
  - » Supports both sniffing/recording devices, such as IDS, and filtering appliances, such as inline firewalls and IPS
  - » Inline and passive modes of operation
  - » Inbound and outbound SSL inspection
- **Policy Configuration:** Fine-grained policy control provides the ability to cut-through non-SSL flows via 7-tuple classification and to control which SSL flows are inspected, passed through or blocked.
- **SSL Session Identification:** The session log provides details of all SSL flows, inspected or not, allowing suspicious trends or patterns of SSL use to be detected.
- **High Availability:** Integrated fail-open hardware, traffic bypass filters, and configurable link state monitoring and mirroring enable guaranteed network availability and network security.
- **Web-based Management:** The Sourcefire SSL Appliance is configured and managed via an SSL-secured web-based graphical user interface, keeping administration simple.
- **Email Alerting:** Logs can be configured to trigger alerts that can be forwarded via email immediately or at intervals to designated network administrators.

To learn more about Sourcefire's award-winning cybersecurity solutions, visit us at www.sourcefire.com or contact Sourcefire or a member of the Sourcefire Global Security Alliance today.

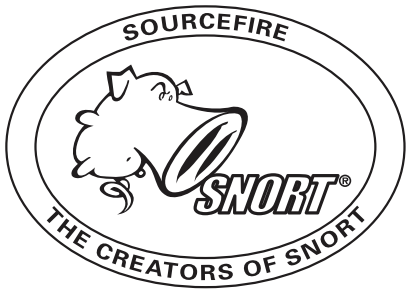
### Product Specifications:

- Up to 1Gbps throughput for SSL traffic
- Less than 40µs cut-through latency
- 1,000,000 simultaneous flows
- 30,000/second SSL flow inspection rate
- 50,000 concurrent SSL flow states
- 2,900/second SSL flow setups/teardowns
- 32,000 traffic diversion policies
- 10,000,000 SSL session log entries
- 2U rack space height
- Available configurations
  - » 4-port (fail-open) 1G copper
  - » 4-port (fail-open) 1G fiber
- Modes of Operation
  - » Passive IDS
  - » Inline IDS
  - » Inline IPS
  - » Inbound inspection
  - » Outbound inspection



SOURCEfire®

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 944
Keysight, EX-1045
Page 944




sourcefire.com | snort.org

©2010 Sourcefire, Inc. All rights reserved. SOURCEFIRE®, Snort®, the Sourcefire logo, the Snort and Pig logo, SOURCEFIRE 3D®, RNA®, SOURCEFIRE DEFENSE CENTER®, SOURCEFIRE RUA®, CLAMAV®, SECURITY FOR THE REAL WORLD™, DAEMONLOGGER™, and certain other trademarks and logos are trademarks or registered trademarks of Sourcefire, Inc. in the United States and other countries. Other company, product and service names may be trademarks or service marks of others.

SOURCEfire®

https://web.archive.org/web/20060503043634/http://eprint.iacr.org:80/2006/136.pdf

# A CHALLENGING BUT FEASIBLE BLOCKWISE-ADAPTIVE CHOSEN-PLAINTEXT ATTACK ON SSL

Gregory V. Bard
*University of Maryland, Department of Mathematics*
*College Park, MD, 20914, USA*
*Email: gregory.bard@ieee.org*

Keywords: Blockwise Adaptive, Chosen Plaintext Attack (CPA), Secure Sockets Layer (SSL), Transport Layer Security (TLS), Cryptanalysis, HTTP-proxy, Initialization Vectors (IV), Cipher Block Chaining (CBC).

Abstract: This paper introduces a chosen-plaintext vulnerability in the Secure Sockets Layer (SSL) and Trasport Layer Security (TLS) protocols which enables recovery of low entropy strings such as can be guessed from a likely set of 2–1000 options. SSL and TLS are widely used for securing communication over the Internet. When utilizing block ciphers for encryption, the SSL and TLS standards mandate the use of the cipher block chaining (CBC) mode of encryption which requires an initialization vector (IV) in order to encrypt. Although the first IV used by SSL is a (pseudo)random string which is generated and shared during the initial handshake phase, subsequent IVs used by SSL are chosen in a deterministic, predictable pattern; in particular, the IV of a message is taken to be the final ciphertext block of the immediately-preceding message, and is therefore known to the adversary.

The one-channel nature of web proxies, anonymizers or Virtual Private Networks (VPNs), results in all Internet traffic from one machine traveling over the same SSL channel. We show this provides a feasible "point of entry" for this attack. Moreover, we show that the location of target data among block boundaries can have a profound impact on the number of guesses required to recover that data, especially in the low-entropy case.

The attack in this paper is an application of the blockwise-adaptive chosen-plaintext attack paradigm, and is the only feasible attack to use this paradigm with a reasonable probability of success. The attack will work for all versions of SSL, and TLS version 1.0. This vulnerability and others are closed in TLS 1.1 (which is still in draft status) and OpenSSL after 0.9.6d. It is hoped this paper will encourage the deprecation of SSL and speed the adoption of OpenSSL or TLS 1.1/1.2 when they are finially released.

## 1 Introduction

This paper outlines a vulnerability of all versions of SSL and also TLS 1.0 by means of a feasible blockwise-adaptive chosen-plaintext attack (BACPA). The attack proceeds as follows. After valuable low-entropy data has been transmitted by the target (e.g. a stock from a list of 2–1000 companies) the adversary inserts plaintext into the communications stream by inducing the target machine to transmit data designed to contain particular byte sequences. Based on the ciphertext values of these known plaintexts, the low-entropy data can be guessed. The probability of success depends on the type of data being targeted, with examples given below. Furthermore, it is shown that the position of this target data within block-boundaries *directly and significantly* impacts the number of guesses required to recover that datum.

The vulnerability described here was closed in TLS 1.1 and OpenSSL after 0.9.6d. We believe all other SSL implementations[1] are vulnerable. Since TLS 1.1 is still in draft status, most TLS deployments use version 1.0, which is vulnerable to this attack. Nonetheless, the main purpose of this paper is not to claim the existence of a major threat to computer security, but rather to disprove the myth that blockwise-adaptive chosen-plaintext attacks are totally infeasible and thus of theoretical interest only. Furthermore, we prove that it is easier to guess the values of low-entropy data when it is divided across block boundaries.[2] It is hoped this paper will highlight some of the disadvantages of Cipher Block Chaining, emphasize the importance of using distinct keys for each source-destination pair, and illustrate the importance of using

[1] There would be exceptions if others copy OpenSSL.
[2] We believe this has not previously been published.

random numbers as initialization vectors.

**The Attack** Modern users are becoming aware that installing software of unknown origin is unsafe. For this reason, and many others, the concept of an "untrusted applet" running in a "sandbox" within the Java Virtual Machine was created (GJSB05). These applets have highly restricted security privileges, but are commonly available on individual and commercial web-sites as utilities and games. If an attacker can manage to get a user to install a Java application (or application in any other language) the attacker can read the user's keystrokes and broadcast them—there is no need for cryptanalysis in this case. However, an untrusted applet does not have this privilege (LÖ97) (GJSB05). It only receives notice ("events") from keystrokes pressed while the applet is on screen and "has the mouse focus." Therefore reading the user's keystrokes while he/she enters valuable data into other applications is impossible. However, one privilege of the untrusted applet is to open a TCP connection to the web server from whence it came (LÖ97). Imagine a user visiting a secure web-site for banking, and later playing a Java applet game. Naturally the socket for the secure transaction and the socket that the applet opens back to its server are unrelated. However, in the special case of a web-anonymizer, HTTP-proxy or virtual private network, all network traffic leaving the user's machine will travel over the same SSL socket[3]. Since they travel over the SSL link and in the same direction, the data in each case will be encrypted with the same secret key. This results in the targeted data, as well as data of the adversary's choosing (the chosen-plaintext), becoming available for use in verifying guesses of the targeted plaintext via Blockwise-Adaptive Chosen-Plaintext attack.

**Blockwise-Adaptive Chosen-Plaintext** BACPA was simultaneously discovered in 2001 by Bellare, Kohno, and Namprempre (BKN02) and by Joux, Martinet, and Valette (JMV02). The BACPA differs from classical chosen plaintext attack (now termed "message-wise" or MCPA) in essentially one detail. In MCPA, the attacker can generate arbitrary messages for the target to encrypt, as part of a sequence of messages. In BACPA, the attacker can generate arbitrary blocks for the target to encrypt, inserted as part of an existing message (i.e., a sequence of blocks). However, four years after its discovery, BACPA has received very little notice outside the cryptographic community, despite several additional papers within it. Bellare, Kohno, and Namprempre (BKN02), in addition to outlining a theoretical model of potential adversary capabilities that is expanded upon in later papers (FJP04) (BT04) (FMP03), loosely describe a general attack on the Secure Shell (SSH) first found by (Dai02). That attack had a success probability[4] of $2^{-12.5}$ or $6.9 \times 10^{-4}$. This paper outlines an attack with a success probability that can approach 100%. We are aware of no other feasible attack under the BACPA model against an existing or proposed protocol.

**The Versions of the SSL Protocol** The Secure Socket Layer (SSL) (FKK96) is currently one of the most widely-used methods for securing communication over the Internet[5]. There are two recent versions, SSL 2.0 (1995) and SSL 3.0 (1996). The successor protocol is Transport Layer Security (TLS), which includes version 1.0 (1999) (DA99), and version 1.1 (DR05) (still in draft status). Also, a version 1.2 has been proposed (DR06). Almost all secure web transactions (i.e. HTTPS) use either TLS 1.0 or SSL 3.0. For simplicity, we refer to all versions of SSL, as well as TLS 1.0, by saying "SSL." OpenSSL is an open source implementation of SSL. Versions of OpenSSL since 0.9.6d (Var) have closed this vulnerability, as has TLS 1.1 (DR05). All SSL deployments other than OpenSSL, including TLS 1.0, are believed to be currently vulnerable to this attack[6].

**Initialization Vectors** Our attack relies on the fact that SSL currently mandates the use of a weak variant of the cipher block chaining (CBC) mode of encryption (KPS02, Chap. 4). CBC mode requires a one-block initialization vector (IV) for each message that is encrypted. In "standard" cryptographic usage of CBC, a fresh, random IV is chosen for each message. In SSL, however, only the *initial* IV is chosen in a (pseudo)random manner; IVs for subsequent messages are simply taken to be the final block of the ciphertext corresponding to the immediately-preceding message. (This process is called "chaining the IVs.") In particular, an attacker may know *in advance* the IV that is going to be used to encrypt the next message. We show that this enables an attacker mounting a chosen-plaintext attack to validate a guess as to the value of a particular plaintext block.

**Consequences** Since the adversary can validate a guess as to the value of a particular plaintext block,

[3] We have recently learned that an e-mail posted to the Certicom TLS mailing list in March of 2002 had also mentioned this possibility, though with fewer details (Res02).

[4] The success probability is stated to be equivalent to waiting for a collision on approximately 25 bits. See the last sentence of Section 3 of (BKN02).

[5] See (KPS02, Chap. 19) for an excellent overview of SSL.

[6] Personal E-mails with E. Rescorla, co-author of the TLS 1.0, 1.1, and 1.2 RFCs (DA99) (DR05) (DR06)

this attack violates the theoretical standard of Left-Or-Right Indistinguishability, whereby no polynomial time adversary can be able to distinguish between the encryption of two messages of his/her own choice, given the ciphertext, (with non-negligible advantage) (GM84) (BDJR97). On the more practical side, it possibly allows an attacker to determine a low-entropy string by repeatedly guessing all possible values for this string until the correct one is identified. Examples of such low-entropy information (2–1000 choices) include names of stocks, cities, users, or even PINs that have been previously encrypted (or, for example, knowing if a stock order is buy, sell, or stop-loss could be valuable information by itself). Given the use of SSL for transmitting exactly this sort of data, we believe this represents a potentially serious (but challenging) attack which should be addressed by any security group selecting an SSL implementation.

## 1.1 Related Work

Essentially this method has been used previously to attack SSH (BKN02) (Dai02). In fact, the flaw attacked there is identical to the flaw attacked here (namely, setting IVs in a predictable way). Little discussion of feasibility or point-of-entry was given, and the probability of success was very low as already stated. On the other hand, that discovery gave birth to the blockwise-adaptive chosen-plaintext world.

Due to the similar structure of SSL and SSH, the related vulnerability in SSL was discovered soon after, independently by Moeller (Moe) and the author of this paper, in late 2002—8 months after the publications of the two original blockwise papers. Moeller's work identifies the attack but does not show how it could be exploited. Moreover, our paper elaborates upon the attack by showing how low-entropy data, in particular, is easy to recover, and provides mechanisms by which to execute the attack. The author of this paper, in 2004, wrote of a similar BACPA attack against SSL but via a much more difficult and complex point-of-entry (Bar04).

The changes to TLS between versions 1.0 and 1.1 were made partially in response to this class of vulnerability[7], as well as that of Vaudenay[8] (DR05).

[7] Attack denoted CBCATT in some TLS documentation (Moe), see also Section 6.2.3.2 of (DA99) (DR05).

[8] Another interesting attack on the use of CBC in SSL, discovered by Vaudenay, relates to the padding of messages, and is unrelated to the vulnerability in this paper (Vau01).

# 2 High-Level Outline

We begin by briefly highlighting the minimal aspects of SSL needed to understand our attack at a high level. A more detailed treatment of the attack (and hence of SSL) is given in Section 3. A good survey of the SSL protocol is given in (KPS02, Chap. 19).

The SSL protocol begins with a handshaking stage during which the parties agree on a protocol version, select cryptographic and (optionally) compression algorithms, perform optional authentication steps, and use public-key mechanisms to share secrets. The shared secrets, which include distinct symmetric keys and IVs for each direction of communication, can then be used for symmetric-key encryption and message authentication. While messages may optionally be compressed before encryption, few SSL implementations do so (KPS02, Chap. 19), (FKK96).

The SSL standard allows for symmetric-key encryption using either block ciphers or stream ciphers. Most implementations utilize block ciphers, and the vulnerability in this paper applies only when block ciphers are used. A block cipher is a keyed, invertible permutation over strings of some fixed length called *blocks*; DES, for example, operates on 64-bit blocks. We write $F_{sk}(X)$ to represent the application of a block cipher using key $sk$ to block $X$. To encrypt messages longer than one block in length, a *mode of encryption* must be used. SSL mandates the cipher block chaining (CBC) mode, which encrypts a message $P = P_1, \ldots, P_\ell$ (where the length of each $P_i$ is the block-length of the cipher) as follows: given some IV denoted $C_0$, compute $C_1, \ldots, C_\ell$ sequentially via:

$$C_i = F_{sk}(P_i \oplus C_{i-1}).$$

In the general case of CBC, the resulting ciphertext is usually taken to be $C_0, \ldots, C_\ell$ although if the receiver already knows $C_0$ then it need not be transmitted. To decrypt, the receiver computes $P_i$ for $i = 1$ to $\ell$ via:

$$P_i = F_{sk}^{-1}(C_i) \oplus C_{i-1}.$$

We note that it is considered "standard" security practice to choose a new, random IV for every message that is encrypted. However, the above definition of CBC does not force this to be the case. As we have mentioned already, SSL chooses all but the initial IV by setting it equal to the final ciphertext block of the preceding encrypted message; this is referred to as "chaining IVs across messages". (Thus, continuing the above example, the IV used for the next message would simply be $C_\ell$.) SSL chooses the initial IV in a pseudorandom fashion which is not important for the purposes of the present attack.

**The attack.** Suppose an adversary who can mount a chosen-plaintext attack wants to verify a guess as to whether some plaintext block has a particular

value. Specifically (continuing the above example), suppose an adversary who has observed the ciphertext $C_0, \ldots, C_\ell$ wants to determine whether plaintext block $P_j$ is equal to $P^*$. Note that the adversary knows the IV (i.e., $C_\ell$) that will be used when encrypting the next message. Consider now what happens if the adversary causes the sender to encrypt a message $P'$ whose initial block $P'_1$ is equal to $C_{j-1} \oplus C_\ell \oplus P^*$. The first ciphertext block $C'_1$ is then:

$$\begin{aligned} C'_1 &= F_{sk}(P'_1 \oplus C_\ell) \\ &= F_{sk}((C_{j-1} \oplus C_\ell \oplus P^*) \oplus C_\ell) \\ &= F_{sk}(P^* \oplus C_{j-1}). \end{aligned}$$

However, we also know that $C_j = F_{sk}(P_j \oplus C_{j-1})$. This implies that $C'_1 = C_j$ if and only if $P_j = P^*$, since $F_{sk}$ is a permutation. In this way, an adversary can verify a guess $P^*$ for the value of any plaintext block $P_j$. In particular, if the adversary knows that $P_j$ is one of two possible values then the adversary can determine the actual value by executing the above attack a single time. Similarly, if the attacker knows that $P_j$ is one of $N$ possible values then by repeating the above attack $N/2$ times (on average) the adversary can determine the actual value of $P_j$. This already violates the standard notions of security for encryption (in terms of Left-Or-Right Indistinguishability, as mentioned in the Introduction on page 3 (GM84) (BDJR97)). Moreover, this implies that an attack of this form can be used to determine the value of a low-entropy string in its entirety. (Note that, in practice, the block of plaintext containing the user's data also likely contains additional information such as headers, etc. However, it is also likely that this additional information is known to the attacker; for example, if the information is fixed padding, then an adversary can learn the format of this data from the web-page source code. We discuss this further in Section 4.)

**Attack requirements.** Focusing specifically on the case of an adversary trying to recover a user's targeted low-entropy data, we briefly highlight the requirements needed for the above-described attack to succeed; in Section 4 we discuss in more detail how these requirements are typically met in practice. First, the attacker must know which plaintext block $j$ contains the desired information. All this means is that the adversary knows the format of the HTTPS transmission being targeted. Second, the adversary must know $C_{j-1}$. However, since the ciphertext travels over the Internet (in the clear!), this is not expected to be difficult. (In fact, if it is assumed difficult to obtain this information then there is little reason to use encryption in the first place.) Third, the adversary must know the value of the IV that is going to be used for the next message. However, we have noted already that because of the way SSL computes IVs, an attacker would actually obtain this information from the last ciphertext block of the previous message. Finally, the adversary must be able to insert a plaintext block of its choice into the first block of the next message to be transmitted. This is the most challenging part of the above attack.

**The scenario.** The mechanism by which the adversary executes the attack is summarized as follows. First, we assume that the user is connected to an HTTP-proxy, web anonymizing service, or virtual private network. This guarantees that all Internet traffic will be routed over one SSL tunnel with a single secret key. Second, we assume the attacker can setup a site under his/her control, and create an applet for the user to open. In particular, applets have features, which will be described below in Section 4, that make them desirable for use. Third, a "reflector" or mechanism for observing the target's ciphertexts is set up. This is needed in all forms of cryptanalysis, so we presume it is available. Fourth, we assume that the user can be induced into opening the applet. This can be in the form of an inviting email which persuades the user to go to the site, or perhaps an email that simply causes the applet to be opened upon viewing, via HTML encoded requests for the applet in the email body, using the <APPLET> tag. We fully acknowledge that these assumptions are non-trivial but they demonstrate that a chosen-plaintext attack is feasible.

Note that untrusted Java applets have very few privileges (GJSB05), but one of those privileges is to open a TCP socket to the server from whence it came (L9̈7). They cannot, for example, simply read all of the user's keystrokes and export them to the adversary, as explained before. Since all outgoing traffic, in the case of a web anonymizer, HTTP-proxy, or virtual private network, travels over the same SSL link, this permits the applet to send and receive arbitrary data—which will become the chosen plaintexts of the attack.

# 3 Attacking SSL

Here, we simply note that there is nothing in the structure of SSL (such as extraneous headers or formatting information) which prevents the attack of the previous section from succeeding.

SSL sits between the Application and Transport layers, and so acts like a Session Layer in the OSI model. As such, SSL receives plaintext from the Application Layer as raw data. This plaintext is fragmented into blocks of length less than or equal to $2^{14}$ bytes. These blocks are optionally (but rarely) com-

pressed[9] and are then processed and sent as follows:

- **Unencrypted Portion :**
  - `message type (8 bits);`
  - `major/minor version number (16 bits);`
  - `length counter (16 bits);`
- **Encrypted Portion :**
  - `plaintext fragment (`$\leq 2^{17}$` bits);`
  - `message authentication code (typically 160 bits);`
  - `padding (0--56 bits; ensures plaintext length is a multiple of block length);`
  - `padding length (8 bits);`

We stress that *the first block of the plaintext is indeed the first block to be encrypted*. In particular, the header information that is prepended to the eventual transmission (i.e., the message type, major/minor version number, etc. . . ). is *not* encrypted. Thus, as long as the adversary can set the first block of the plaintext fragment to some desired value (as discussed in the previous section), that block will be encrypted first and the attack will succeed.

We note that in SSH, some header data is prepended to the plaintext *before* encryption. This makes an attack such as the one outlined here more difficult in the context of SSH (BKN02) (Dai02), since the adversary no longer has control over the first block of the plaintext that is eventually encrypted. Although it may be possible to work around these constraints (see (BKN02)), the attack is more difficult against SSH than it is against SSL (demonstrated by a success rate of $2^{-12.5}$ as compared to rates as high as 100% under certain circumstances).

# 4 Feasibility

Several challenges must be surmounted before an adversary can successfully perform the attack that has been outlined here. The necessary steps to meet these challenges are listed below.

**Using one common key.** First and foremost, the data to be learned and the test cases must be encrypted with the same key. Furthermore, since the key for the block cipher is chosen at random each time an SSL socket is created, the data to be learned and the test cases must be transmitted through the same tunnel, and in the same direction. Luckily enough, when a web-tunneling service like an HTTP-proxy, or a VPN is used, all the out-going data travels over one SSL link, regardless of destination, and thus is encrypted with the same secret key.

**Learning the plaintext format.** Despite the length of a plaintext message, there are times when only a very small sequence of bytes is of critical importance. For example, a stock choice, a destination city for a geospatial inquiry, a user name, or something where only 2–1000 choices are considered likely, and known in advance (i.e., a "low-entropy string" in our terminology).

We have mentioned earlier that the adversary must somehow know which block of the plaintext contains the data of interest. Note, however, this is easily done by reading the source files for the pages that are used in sending the data. Discerning the format merely requires knowledge of HTTP, HTML, and CGI, and perhaps Javascript. Commonly available browsers have a "show page source" command, which displays the page's HTML source code. Both the "form elements" which compile the user's data, as well as the optional Javascript code which would verify its format, would thus be available to an attacker. While Javascript can alter HTML code, for example, to add hidden form elements, these would be visible in the Javascript code. The attacker need only read this code and the format is trivially derived.

**Ensuring that the adversary's chosen plaintext block is encrypted first.** It is essential that the chosen plaintext, namely $P_i = P^* \oplus C_\ell \oplus C_{j-1}$, be the first encrypted block of the SSL datagram inside which it is found. However, this is easy, as we argue now.

Data is submitted to the SSL layer in the form of application-level messages, which are first aggregated into blocks of (at most) $2^{14}$ bytes in length. SSL does not respect message boundaries. If more than $2^{14}$ bytes are submitted, additional blocks are created; if several application level messages are submitted, they are concatenated in the buffer. However, in the absence of these two conditions, the data is encrypted and transmitted to the TCP layer immediately. Therefore, short messages would be encrypted and transmitted immediately. The structure of SSL packets guarantees, in the absence of concatenation, that the first block of the application message will be the first encrypted block of the SSL datagram.

Concatenation only occurs when two messages arrive at the SSL layer simultaneously. However, since the packet source (the applet or game) is under the control of the adversary, the timing could be adjusted to guarantee that two packets never arrive during the required interval. Of course, packets could arrive from some other user activity or application. These timing failures can be avoided probabilistically by repeating each guess several times. Finally it should be noted that in the event concatenation does occur, if the game packet is the first packet, with other packets

[9] Our attacks do not apply when compression is used; however, we note that it is rarely used.

appended, no disadvantage is caused for the attack. Nonetheless, we acknowledge the timing difficulties.

**Encrypting Chosen Plaintext.** The attacker must force the encryption of particular data in any chosen plaintext attack, in this case based upon the previous ciphertext block, the guess of the targeted plaintext, and the ciphertext previous to the data to be guessed. Once this block is calculated, it must be inserted into the "plaintext stream."

Since our scenario of a user connecting via an HTTP-proxy or a VPN involves *all network traffic flowing to/from the target machine being encrypted by the same SSL connection,* with a distinct key for each direction, the attacker need only cause the chosen plaintext data to appear in an application-level datagram, traveling in the same direction as the targeted plaintext. However, there are other considerations.

- **The data must start the datagram.** Once the immediately previous ciphertext block is known, the very next plaintext block must contain the attacker's plaintext. There must be no blocks in between, otherwise our previous formulas fail. Therefore, it is essential that the test block (first block of new plaintext) be the first block of the datagram.
- **The data must be tolerated.** The data must not result in a crashed application, a dropped connection or error messages that would alert the user. Sending arbitrary bytes to an application via a datagram would almost surely violate that application protocol's datagram formatting rules. The data must "blend in" to the data that would have been otherwise sent.[10] For example, if the application has a header for each datagram, then its unlikely that the cryptanalysis blocks, being the first blocks of the datagram, will conform to that header's definition.
- **The application must anticipate many datagrams.** Since only one guess can be validated per packet, the application must anticipate a moderate number of packets (on the order of 1–500 for a guess among a set of 2–1000 choices).
- **Background Traffic** The attack will also be less likely detected if there is background traffic to/from the applet when the attack is not running.

Given these four requirements, it seems desirable to write one's own application for the attack, rather than find an application that can tolerate the above. In particular, if the applet is programmed to search for a 128-bit "magic number" in each datagram, then it can discard all the data before and including that number. This way the cryptanalysis blocks can be safely ignored. The chances of the magic number occurring accidently are almost infinitesimal.

[10]Special thanks to an anonymous referee for pointing out several challenges in this regard.

**Providing feedback to the Applet.** Note that the adversary needs feedback from the reflector to the applet site (in particular, to inform it of the ciphertext blocks $C_{j-1}, C_j, C_\ell$) in order to perform the attack. It is not expected to be difficult for an adversary's reflector to obtain these ciphertexts (after all, they are traveling on the Internet), but the information must somehow be communicated to the applet site.

There are two principal avenues through which this communication could occur. First, if the attack is being performed in a wireless environment, setting a wireless device to promiscuous mode is attractive. However, many systems use WEP (Wired Equivalent Privacy) though certainly not all (including, for example, the University of Maryland and American University campus networks, which are unencrypted).

Alternatively, if the adversary has access to the same Ethernet network on his/her own machine that the user is connected to, then the following can be executed. Perhaps a subordinate employee will use his/her private laptop with tcpdump and a special feedback script acting as a reflector, to capture his/her employer's SSL traffic. The game applet's server would only need to know where the feedback daemon is, receive the ciphertexts, and use it to compute new plaintexts. The search for the daemon could be through the use of a throw-away DNS address, or by hard-coding the IP address.

On the one hand this could be detected by an Intrusion Detection System as unauthorized traffic, but on the other, steganography could be used to embed the data in legitimate communications. Alternatively, since few Intrusion Detection Systems would be this sensitive, the reflector could simply pass the packets back in the clear.

**Synchronization.** The demands of synchronization are important since there can be no transmitted data between the "last ciphertext block" and the encrypted attacking plaintext. There may be many ways to achieve this, but the game/applet server could signal that it is ready by sending a special packet. The reflector responds with the most recently transmitted ciphertext, which hopefully travels with minimal delay to the applet. Upon receipt of this, the applet transmits a packet, with or without game data. The first 8 or 16 bytes of the packets (one encrypted word) is not game data, but the chosen-plaintext. If no data (from any application) was transmitted on the tunnel in this time, then this stage of the attack succeeds and one guess can be checked. If data was transmitted on the tunnel in this time, then the plaintext loses its meaning in this context, but no harm is done and the attack can be attempted again. Since the Poisson Distribution is known to be a good model of user packet generation during web-browsing and other non-streaming Internet tasks, large gaps between packets will be ex-

pected. This is especially true if the game is interesting and the user is not surfing other sites while playing it.

If the user will play for as little as 15 minutes, many packets can make the round trip in this time. Since the "go signal", the reflector's "last seen ciphertext", and the "guess" must all be exchanged, naturally with acknowledgments (since we are using TCP), six round-trip times are required. Rarely is a round-trip more than 250 ms, so each "guess" takes 1.5 seconds, or 40 guesses can be made per minute. This comes to 600 guesses per short 15 minute game. Some of these guesses will be clobbered by traffic from other sockets, but others should succeed. Even if only one-third of the packets get through (which is very pessimistic), a choice from a list of 200 options will be found with absolute certainty, and a choice from a list of 10,000 options (e.g. a four digit PIN number) will be found with a probability of 2%.

**Summary** The requirements listed in this section are non-trivial. The timing and other network packet structure considerations will result in some guesses failing to serve their purpose. But we believe that we have demonstrated the potential feasibility of this sort of attack. This further demonstrates that BACPA is of more than purely theoretical interest.

## 5 Recovering PIN's

Here we show how even a moderately small entropy string ($\leq$10,000 choices) can be easy to recover due to segmentation that occurs when the target data falls on a block boundary. (We assume throughout this section that the data surrounding the target information is known; see above). Since it is now demonstrated that the adversary has the ability to verify guesses of plaintext blocks, one can imagine that an adversary can attempt to guess the value of a valuable target low-entropy string either by exhaustive search (in the case of Personal Identification Numbers or PINs) or by more efficient context-specific schemes (using dictionary based attacks versus passwords, for example).

If we assume for simplicity that the data is chosen uniformly from a space of size $S$, then the expected number of guesses needed before determining the string is $S/2$. (Note that in the case of passwords chosen by a user, the entropy is likely to be much lower than would be indicated by the length of the password alone. In particular, an 8-character password model typically has entropy much lower than 64 bits). For example, a 4-digit PIN can be determined with (on average) 5,000 guesses. If only 100 guesses can be made, the probability of success is 1%, which is low, but certainly represents a feasible attack.

**Block Cipher Choices:** Recall that AES was not in existence when SSL 3.0 was originally specified (1996), nor was it finalized when TLS 1.0 was released (1999). Both SSL 3.0 and TLS 1.0, the standardized versions now available, do not include AES in their specification (FKK96) (DA99). Likewise, TLS 1.1 does not have AES as a built-in option, but it is available via some extension documents (e.g., (MR06)). Finally TLS 1.2 will have AES built in, as this was one of two principle changes between 1.1 and 1.2, the other relating to the choice of hash functions (DR06). Therefore, essentially all SSL/TLS transactions at this time use DES or 3-DES, both with a 64-bit plaintext and ciphertext block length (though with an effective key size of 56 or 112 bits).

**Split Target Data:** However, assume 3-DES is being used as the block-cipher in the SSL transaction. Then the plaintext blocks are 64 bits or 8 bytes. There is a 12.5% probability that the four bytes of pin-data will be divided exactly in the middle of a block boundary.[11] *Since each block can be guessed independently*, 50 guesses are required for the 100 options on the left, and 50 guesses are required for the 100 options on the right. This use of 100 guesses results in a 25% chance of success, since both halves must be correct—rather than a mere 2% above. Alternatively, fewer guesses can be made if a lower probability of success is tolerated. Even a 3-1 or 1-3 split results in a major distortion of the number of required guesses. And there is a 37.5% chance that such a split (1-3, 2-2 or 3-1) will occur. See the Appendix, in the full version of this paper, for a discussion of applying this technique to passwords.

One should be careful to distinguish forms where passwords are entered all at once, and then submitted via SSL, from the normal SSH method where passwords are entered one keystroke at a time. In the latter case, this trick of breaking up the password would not work—a single byte is never broken up [12].

## 6 Solutions

It has been noted that TLS 1.1 and OpenSSL (after 0.9.6d) are not vulnerable to this attack, for reasons detailed below.

**TLS 1.1 and Explicit IVs** The TLS 1.1 protocol (DR05) fixes this vulnerability by using explicit IVs.

[11] One must assume that enough traffic has gone by that the "indentation" or "offset" within the stream of the valuable data is unpredictable, and therefore can be treated as a uniformly distributed random variable.

[12] Personal E-mails with E. Rescorla, co-author of the TLS 1.0, 1.1, and 1.2 RFCs (DA99) (DR05) (DR06)

That is to say, each message has one more ciphertext block than plaintext blocks. This first ciphertext block is the IV, determined as a (pseudo)random number. As we have stated several times already, this is the accepted way to encrypt using CBC. Since the attacker does not know this value in advance, this attack cannot be executed. Using the formula for CBC, it is easy to see that having an Explicit IV is identical to adding an additional plaintext block of all zeroes to the start of each message, which the receiver knows to discard, and chaining the IVs from message to message (Moe).

If generating truly random bits is a concern (say, for reasons of efficiency), it is easy to generate a pseudorandom IV in any of a number of ways. For example, instead of simply using $C_\ell$ (i.e., the last block of the preceding ciphertext) as the IV, the protocol could use $H(C_\ell|sk)$ where $sk$ is the shared secret key used for encryption and $H$ is a cryptographic hash function.

**Single Block Nonces** This solution is mentioned because it can be used in other applications that use CBC, to protect them from BACPA, and helps to explain the solution used by the OpenSSL community. As was noted previously, the adversary's guess must be the first block of each message. Introducing a one block nonce—which would always be discarded before reading the message—would make this impossible. The nonce does not even need to be random. Suppose the nonce was always the all-zero string, and the previous message ended on block $C_i$. Then the adversary will submit a block $P_{i+2}$ based on a guess $G$ for block $g$. This will be output in block $i+2$, where $i+1$ is the nonce. Therefore the submitted plaintext, from the attack formula given previously would be

$$P_{i+2} = C_{i+1} \oplus C_{g-1} \oplus G$$

But $C_{i+1}$ has not been transmitted yet. It equals

$$C_{i+1} = F_{sk}(C_i \oplus 0000 \cdots 0)$$

Therefore, $C_{i+1}$ can only be determined if $F_{sk}(C_i)$ has been calculated before, or the adversary guesses the $F_{sk}(C_i)$. Since $F_{sk}(\cdot)$ is a function family believed to be pseudorandom, this guess will be correct with negligible probability. Likewise, since the space of all possible ciphertexts is $2^{64}$ or $2^{128}$, the probability of a repetition is low. As mentioned earlier, in the case of the all zero plaintext, this is identical to explicit IVs. The TLS specification for version 1.1 allows for this solution, in addition to Explicit IVs, but recommends that the nonce be pseudorandom, generated for each message independently (DR05).

**OpenSSL and the Empty Message** A slight variation of the above is used by OpenSSL after version 0.9.6d (Var). An empty message, which consists of no plaintext, but only padding and a hash, is prepended to each set of messages before encryption. The "extra parts", namely the padding and hash, are encrypted, and so form the throw-away blocks similar to the nonce above. Since the adversary's chosen blocks are no longer the first to be encrypted, this attack becomes impossible. What is interesting about this solution is that it does not require any major changes at all to the SSL standard, and fulfills the present definition of SSL as written. (This implies only sites, not customers, need change their software). The only point to mention is that some SSL clients will declare an error if an empty message is received. This error message need only be suppressed, which is a minor change.

**Compression** Note that an immediate way to prevent the attack suggested here is to turn compression on (as we have noted, an attack of the sort suggested here is much more difficult — if not impossible — if compression is used). However, this requires that peers only communicate with others who also use compression (or else an adversary connecting to the honest party could mount a "chosen-protocol attack" in which they claim to be unable to use compression). On the other hand, blocking users not configured for compression would limit inter-operability with deployed versions of SSL.

# 7 Conclusions

The attack presented here is not so easy that it can be done on the spur of the moment by the typical hacker. However, while the attack is challenging to carry out, the success probability and relatively low numbers of datagrams required should be sufficient to motivate the SSL community to migrate away from TLS 1.0 and SSL, to OpenSSL after 0.9.6d, or TLS 1.1/1.2 when they are finally released. Moreover, by demonstrating the existence of this attack on a real-world protocol, which corresponds to the theoretical definition of blockwise-adaptive chosen-plaintext attack, we prove that the BACPA model is not sterile, but is useful for modeling adversarial capabilities.

Moreover it is hoped that this work will take a step toward opening the dialog between protocol designers and theoretical cryptographers, and stimulate discussion between these two camps which are otherwise independent. Finally, there are other uses of CBC similar to that of SSL, and this attack shows that those applications should also use explicit IVs or another solution listed here (e.g. Datagram Transport Layer Security or DTLS (MR04)).

## 8 Acknowledgments

Thanks to Prof. Jonathan Katz for suggesting the problem, for helpful discussions, and for substantial help editing this document. Thanks also to Ruggero Morselli, Patrick Studdard, Radostina Koleva, Zhongchao Yu, Susan Schmoyer and Prof. Lawrence Washington (all of the University of Maryland), for proofreading early versions of this paper. Valuable feedback was received from Daniel Brown of CertiCom, Eran Tromer of the Weizmann Institute, Prof. Bodo Moeller of UC Berkeley, and Jack Lloyd of randombit.net, about an earlier paper outlining a different attack on this same SSL vulnerability (Bar04). Most importantly we would like to thank Eric Rescorla, coauthor of the TLS RFCs (DA99) (DR05), (DR06), for his extended correspondences via email that helped this work immensely.

## REFERENCES

G. Bard. The vulnerability of ssl to chosen-plaintext attack. Cryptology ePrint Archive, Report 2004/111, 2004. `http://eprint.iacr.org/`.

M. Bellare, A. Boldyreva, L. Knudsen, and C. Namprempre. On-line ciphers and the hash-cbc construction. In *Lecture Notes in Computer Science*. Advances in Cryptology— CRYPTO'01, Springer-Verlag, 2001.

M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption: Analysis of the des modes of operation. In *Symposium on the Foundations of Computer Science (FOCS'97)*. IEEE, 1997.

M. Bellare, T. Kohno, and C. Namprempre. Provably fixing the ssh binary packet protocol. In *Conference on Computer and Communications Security (CCS'02)*. ACM, 2002.

M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *Lecture Notes in Computer Science*. Advances in Cryptology— ASIACRYPT'00, Springer-Verlag, 2000.

A. Boldyreva and N. Taesombut. On-line encryption schemes: New security notions and constructions. In *Cryptographer's Track*. RSA Conference, 2004.

T. Dierks and C. Allen. The tls protocol, version 1.0. Technical Report RFC 2246, Internet Engineering Task Force, 1999.

W. Dai. An attack against ssh2 protocol. Email to the ietf-ssh@netbsd.org email list, February 2002.

T. Dierks and E. Rescorla. The tls protocol, version 1.1. Technical Report RFC 2246-bis-11, Internet Engineering Task Force, 2005.

T. Dierks and E. Rescorla. The tls protocol, version 1.2. Technical Report RFC 4346-bis-00, Internet Engineering Task Force, 2006.

M. Dworkin. Recommendation for block cipher modes of operation: Methods and techniques. Technical Report NIST Special Publication 800-38A, National Institute of Science and Technology, 2001.

M. Dworkin. Recommendation for block cipher modes of operation: The rmac authentication mode, methods and techniques. Technical Report NIST Special Publication 800-38B, National Institute of Science and Technology, 2002.

P. Fouque, A. Joux, and G. Poupard. Blockwise adversarial model for on-line ciphers and symmetric encryption schemes. In *Lecture Notes in Computer Science*. Advances in Cryptology— SAC'04, Springer-Verlag, 2004.

A. Freier, P. Karlton, and P. Kocher. The ssl protocol, version 3.0. Technical report, Transport Layer Security Working Group Internet Draft, 1996.

P. Fouque, G. Martinet, and G. Poupard. Practical symmetric on-line encryption. In *Lecture Notes in Computer Science*. Advances in Cryptology— FSE'03, Springer-Verlag, 2003.

V. Gligor and P. Donescu. Fast encryption and authentication: Xcbc encryption and xecb authentication modes. In *2nd NIST Workshop on AES Modes of Operation*. National Institute of Science and Technology, 2001.

J. Gosling, B. Joy, G. Steele, and G. Bracha. *The Java(TM) Language Specification*. Addison-Wesley Professional, third edition, 2005.

S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 1984.

A. Joux, G. Martinet, and F. Valette. Blockwise-adaptive attackers: Revisiting the (in)security of some provably secure encryption models: Cbc, gem, iacbc. In *Lecture Notes in Computer Science*. Advances in Cryptology— CRYPTO'02, Springer-Verlag, 2002.

L. Knudsen. Block chaining modes of operation. In *Symmetric Key Block Cipher Modes of Operation Workshop*. National Institute of Science and Technology, 2000.

C. Kaufman, R. Perlman, and M. Speciner. *Network Security: Private Communication in a Public World*. Prentice Hall, second edition, 2002.

H. Krawczyk. The order of encryption and authentication for protecting communications (or: How secure is ssl?). In *Lecture Notes in Computer Science*. Advances in Cryptology— CRYPTO'01, Springer-Verlag, 2001.

S. Löffler. Using flows for analysis and measurement of internet traffic. Master's thesis, Institute of Communication Networks and Computer Engineering of the University of Stuttgart, 1997. `http://www.mathematik.uni-stuttgart.de/~floeff/diplom/report/node62.htm%l`.

H. Lipmaa, P. Rogaway, and D. Wagner. Comments to nist concerning aes modes of operation: Ctr-mode encryption. In *Symmetric Key Block Cipher Modes of Operation Workshop*. National Institute of Science and Technology, 2000.

B. Moeller. Security of cbc ciphersuites in ssl (tls problems and counter-measures. Posting on the Open SSL Project's website. http://www.openssl.org/~bodo/tls-cbc.txt.

N. Modadugu and E. Rescorla. The design and implementation of datagram tls. In *Network Distributed System Security Conference*, 2004.

N. Modadugu and E. Rescorla. Aes counter mode cipher suites for tls and dtls. Technical report, Internet Engineering Task Force, 2006.

E. Rescorla. [ietf-tls] re: Rfc 2246-bis open issues. Email to the ietf-tls@lists.certicom.com email list, March 2002. http://www.imc.org/ietf-tls/mail-archive/msg03341.html.

Various. Various documents at the Open SSL web-site. http://www.openssl.org/.

S. Vaudenay. Security flaw induced by cbc padding applications to ssl, ipsec, wtls, .... In *Lecture Notes in Computer Science*. Advances in Cryptology— EUROCRYPT'02, Springer-Verlag, 2001.

# A General Problems with CBC

The Cipher Block Chaining (CBC) mode of encryption was first proposed as a mode for DES, the Data Encryption Standard (BDJR97). However, in the three decades that have passed since that time, much research has taken place in both adversarial modeling and modes of encryption (BBKN01) (Dwo01) (Dwo02) (FMP03) (GD01) (Knu00) (LRW00).

A mode of encryption is an algorithm for defining how the block cipher will be used to produce ciphertexts when the plaintext is of length longer than one block. For example, CBC has the formula $C_i = F_{sk}(C_{i-1} \oplus P_i)$ where $C_0$ is an initialization vector, and Counter Mode (CTR) has the formula $C_i = F_{sk}(i+i_0) \oplus P_i$, where $i_0$ is an initialization vector (Knu00) (LRW00).

We note the following disadvantages of CBC.

- CBC is vulnerable to blockwise-adaptive chosen-plaintext attack, while CTR is not.
- An error in one block of CBC renders unreadable the remainder of the message, while an error in one block of CTR only renders that block unreadable.
- CBC cannot be parallelized, as can CTR. In CBC each block depends on the encryption of the block before it. In CTR, each block is encrypted independently.
- CBC is subject to the padding attack of Vaudenay (Vau01), but this is avoidable if one pads according to the algorithm given in that paper. In CTR, one can pad arbitrarily and have the padding length as an extra plaintext block at the end.
- CBC offers no protection versus Chosen Ciphertext Attack (CCA), as would HPCBC, XCBC, or OCB (BBKN01) (GD01). However, Counter Mode also offers no CCA protection.
- An initialization vector need only be calculated once in CTR mode, not per message as in CBC. The counter $i$ can be statefully maintained, and so will not repeat until $2^{64}$ or $2^{128}$ blocks.
- While this does not necessarily apply to SSL, the $F_{sk}(i+i_0)$ can be pre-computed, leaving only XOR operations with the plaintext to compute the ciphertext. If data needs to be transmitted only occasionally, but urgently when ready, this can be an advantage.

Since SSL uses Message Authentication Code (MAC) algorithms for the datagrams, the maliability attacks and other CCA attacks that plague CTR mode are of no interest. Therefore it is clear that CTR and not CBC would be a better choice for future versions of TLS and SSL, as CTR has advantages and no disadvantages over CBC (LRW00).

**Other Problems with SSL's Encryption:** We note that the mode of encryption should probably be changed to address other concerns as well. For example, it is well-known that applying a message authentication code to the ciphertext itself *after* encryption is preferable to applying it to the message *before* encryption (BN00) (Kra01). Currently, SSL does the latter rather than the former.

# B The Application of Splitting Blocks to Guessing Passwords

While the attack presented in this paper really can only provide for up to about 1000 guesses at best (a long game with little background traffic), under ideal conditions, other chosen plaintext attacks might allow for several more. Therefore it is interesting to point out the effect of splitting upon passwords. If the printable ASCII character set of 95 choices is used, and the passwords are 8 bytes long, and nearly randomly chosen (very generous assumptions), then there are $95^8 = 6.6 \times 10^{15} = 2^{52.6}$ possible passwords, and $2^{51.6} \approx 3.4 \times 10^{15}$ guesses would be required in expectation.

However, the probability of the password not being broken in two is 12.5%. Even with AES and 16-byte blocks, $\frac{7}{16} = 43.8\%$ will be broken into pieces. If divided down the center, (regardless of 64-bit or 128-bit blocks) the number of guesses expected would be

$$\frac{2^{\frac{52.6}{2}} + 2^{\frac{52.6}{2}}}{2} = 2^{26.3} = 8.26 \times 10^7$$

Since $3.4 \times 10^{15}$ guesses are expected in the unsplit case, and $8.3 \times 10^7$ in the split case, the attack becomes $4.1 \times 10^7$ times faster when the password is split. While our present attack scenario is not suited for anywhere near this number of guesses, there may be similar scenarios which can tolerate a few tens of thousands guesses, and if a few thousand users are targeted, then at least one password recovery would be expected, if not more.

Network Working Group
Request for Comments: 4346
Obsoletes: 2246
Category: Standards Track

T. Dierks
Independent
E. Rescorla
RTFM, Inc.

April 2006

# The Transport Layer Security (TLS) Protocol Version 1.1

## Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Copyright Notice

Copyright (C) The Internet Society (2006).

## Abstract

This document specifies Version 1.1 of the Transport Layer Security (TLS) protocol. The TLS protocol provides communications security over the Internet. The protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery.



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 957
Keysight, EX-1045
Page 957

Table of Contents


1. Introduction ....4
   1.1. Differences from TLS 1.0 ....5
   1.2. Requirements Terminology ....5
2. Goals ....5
3. Goals of This Document ....6
4. Presentation Language ....6
   4.1. Basic Block Size ....7
   4.2. Miscellaneous ....7
   4.3. Vectors ....7
   4.4. Numbers ....8
   4.5. Enumerateds ....8
   4.6. Constructed Types ....9
        4.6.1. Variants ....10
   4.7. Cryptographic Attributes ....11
   4.8. Constants ....12
5. HMAC and the Pseudorandom Function ....12
6. The TLS Record Protocol ....14
   6.1. Connection States ....15
   6.2. Record layer ....17
        6.2.1. Fragmentation ....17
        6.2.2. Record Compression and Decompression ....19
        6.2.3. Record Payload Protection ....19
               6.2.3.1. Null or Standard Stream Cipher ....20
               6.2.3.2. CBC Block Cipher ....21
   6.3. Key Calculation ....24
7. The TLS Handshaking Protocols ....24
   7.1. Change Cipher Spec Protocol ....25
   7.2. Alert Protocol ....26
        7.2.1. Closure Alerts ....27
        7.2.2. Error Alerts ....28
   7.3. Handshake Protocol Overview ....31
   7.4. Handshake Protocol ....34
        7.4.1. Hello Messages ....35
               7.4.1.1. Hello request ....35
               7.4.1.2. Client Hello ....36
               7.4.1.3. Server Hello ....39
        7.4.2. Server Certificate ....40
        7.4.3. Server Key Exchange Message ....42
        7.4.4. Certificate request ....44
        7.4.5. Server Hello Done ....46
        7.4.6. Client certificate ....46
        7.4.7. Client Key Exchange Message ....47
               7.4.7.1. RSA Encrypted Premaster Secret Message ....47
               7.4.7.2. Client Diffie-Hellman Public Value ....50
        7.4.8. Certificate verify ....50
        7.4.9. Finished ....51







8. Cryptographic Computations ....52
   8.1. Computing the Master Secret ....52
        8.1.1. RSA ....53
        8.1.2. Diffie-Hellman ....53
9. Mandatory Cipher Suites ....53
10. Application Data Protocol ....53
11. Security Considerations ....53


https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 958
Keysight, EX-1045
Page 958

12. IANA Considerations .....54
A. Appendix - Protocol constant values .....55
A.1. Record layer .....55
A.2. Change cipher specs message .....56
A.3. Alert messages .....56
A.4. Handshake protocol .....57
A.4.1. Hello messages .....57
A.4.2. Server authentication and key exchange messages ....58
A.4.3. Client authentication and key exchange messages ....59
A.4.4.Handshake finalization message .....60
A.5. The CipherSuite .....60
A.6. The Security Parameters .....63
B. Appendix - Glossary .....64
C. Appendix - CipherSuite definitions .....68
D. Appendix - Implementation Notes .....69
D.1 Random Number Generation and Seeding .....70
D.2 Certificates and authentication .....70
D.3 CipherSuites .....70
E. Appendix - Backward Compatibility With SSL .....71
E.1. Version 2 client hello .....72
E.2. Avoiding man-in-the-middle version rollback .....74
F. Appendix - Security analysis .....74
F.1. Handshake protocol .....74
F.1.1. Authentication and key exchange .....74
F.1.1.1. Anonymous key exchange .....75
F.1.1.2. RSA key exchange and authentication .....75
F.1.1.3. Diffie-Hellman key exchange with authentication ..76
F.1.2. Version rollback attacks .....77
F.1.3. Detecting attacks against the handshake protocol ...77
F.1.4. Resuming sessions .....78
F.1.5. MD5 and SHA .....78
F.2. Protecting application data .....78
F.3. Explicit IVs .....79
F.4 Security of Composite Cipher Modes .....79
F.5 Denial of Service .....80
F.6. Final notes .....80
Normative References .....81
Informative References .....82





## 1. Introduction

The primary goal of the TLS Protocol is to provide privacy and data integrity between two communicating applications. The protocol is composed of two layers: the TLS Record Protocol and the TLS Handshake Protocol. At the lowest level, layered on top of some reliable transport protocol (e.g., TCP[TCP]), is the TLS Record Protocol. The TLS Record Protocol provides connection security that has two basic properties:

- The connection is private. Symmetric cryptography is used for data encryption (e.g., DES [DES], RC4 [SCH] etc.). The keys for this symmetric encryption are generated uniquely for each connection and are based on a secret negotiated by another

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

protocol (such as the TLS Handshake Protocol). The Record Protocol can also be used without encryption.

- The connection is reliable. Message transport includes a message integrity check using a keyed MAC. Secure hash functions (e.g., SHA, MD5, etc.) are used for MAC computations. The Record Protocol can operate without a MAC, but is generally only used in this mode while another protocol is using the Record Protocol as a transport for negotiating security parameters.

The TLS Record Protocol is used for encapsulation of various higher-level protocols. One such encapsulated protocol, the TLS Handshake Protocol, allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or receives its first byte of data. The TLS Handshake Protocol provides connection security that has three basic properties:

- The peer's identity can be authenticated using asymmetric, or public key, cryptography (e.g., RSA [RSA], DSS [DSS], etc.). This authentication can be made optional, but is generally required for at least one of the peers.

- The negotiation of a shared secret is secure: the negotiated secret is unavailable to eavesdroppers, and for any authenticated connection the secret cannot be obtained, even by an attacker who can place himself in the middle of the connection.

- The negotiation is reliable: no attacker can modify the negotiation communication without being detected by the parties to the communication.

One advantage of TLS is that it is application protocol independent. Higher level protocols can layer on top of the TLS Protocol





transparently. The TLS standard, however, does not specify how protocols add security with TLS; the decisions on how to initiate TLS handshaking and how to interpret the authentication certificates exchanged are left to the judgment of the designers and implementors of protocols that run on top of TLS.

### 1.1. Differences from TLS 1.0

This document is a revision of the TLS 1.0 [TLS1.0] protocol, and contains some small security improvements, clarifications, and editorial improvements. The major changes are:

- The implicit Initialization Vector (IV) is replaced with an explicit IV to protect against CBC attacks [CBCATT].

- Handling of padding errors is changed to use the bad_record_mac alert rather than the decryption_failed alert to protect against CBC attacks.

- IANA registries are defined for protocol parameters.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

- Premature closes no longer cause a session to be nonresumable.

- Additional informational notes were added for various new attacks on TLS.

In addition, a number of minor clarifications and editorial improvements were made.

### 1.2. Requirements Terminology

In this document, the keywords "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT" and "MAY" are to be interpreted as described in RFC 2119 [REQ].

## 2. Goals

The goals of TLS Protocol, in order of their priority, are as follows:

1. Cryptographic security: TLS should be used to establish a secure connection between two parties.

2. Interoperability: Independent programmers should be able to develop applications utilizing TLS that can successfully exchange cryptographic parameters without knowledge of one another's code.





3. Extensibility: TLS seeks to provide a framework into which new public key and bulk encryption methods can be incorporated as necessary. This will also accomplish two sub-goals: preventing the need to create a new protocol (and risking the introduction of possible new weaknesses) and avoiding the need to implement an entire new security library.

4. Relative efficiency: Cryptographic operations tend to be highly CPU intensive, particularly public key operations. For this reason, the TLS protocol has incorporated an optional session caching scheme to reduce the number of connections that need to be established from scratch. Additionally, care has been taken to reduce network activity.

## 3. Goals of This Document

This document and the TLS protocol itself are based on the SSL 3.0 Protocol Specification as published by Netscape. The differences between this protocol and SSL 3.0 are not dramatic, but they are significant enough that TLS 1.1, TLS 1.0, and SSL 3.0 do not interoperate (although each protocol incorporates a mechanism by which an implementation can back down prior versions). This document is intended primarily for readers who will be implementing the protocol and for those doing cryptographic analysis of it. The specification has been written with this in mind, and it is intended to reflect the needs of those two groups. For that reason, many of the algorithm-dependent data structures and rules are included in the body of the text (as opposed to in an appendix), providing easier

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 961
Keysight, EX-1045
Page 961

access to them.

This document is not intended to supply any details of service definition or of interface definition, although it does cover select areas of policy as they are required for the maintenance of solid security.

## 4. Presentation Language

This document deals with the formatting of data in an external representation. The following very basic and somewhat casually defined presentation syntax will be used. The syntax draws from several sources in its structure. Although it resembles the programming language "C" in its syntax and XDR [XDR] in both its syntax and intent, it would be risky to draw too many parallels. The purpose of this presentation language is to document TLS only; it has no general application beyond that particular goal.





### 4.1. Basic Block Size

The representation of all data items is explicitly specified. The basic data block size is one byte (i.e., 8 bits). Multiple byte data items are concatenations of bytes, from left to right, from top to bottom. From the bytestream, a multi-byte item (a numeric in the example) is formed (using C notation) by:

```
value = (byte[0] << 8*(n-1)) | (byte[1] << 8*(n-2)) |
        ... | byte[n-1];
```

This byte ordering for multi-byte values is the commonplace network byte order or big endian format.

### 4.2. Miscellaneous

Comments begin with "/*" and end with "*/".

Optional components are denoted by enclosing them in "[[ ]]" double brackets.

Single-byte entities containing uninterpreted data are of type opaque.

### 4.3. Vectors

A vector (single dimensioned array) is a stream of homogeneous data elements. The size of the vector may be specified at documentation time or left unspecified until runtime. In either case, the length declares the number of bytes, not the number of elements, in the vector. The syntax for specifying a new type, T', that is a fixed-length vector of type T is

```
T T'[n];
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

Here, T' occupies n bytes in the data stream, where n is a multiple of the size of T. The length of the vector is not included in the encoded stream.

In the following example, Datum is defined to be three consecutive bytes that the protocol does not interpret, while Data is three consecutive Datum, consuming a total of nine bytes.

```
opaque Datum[3];      /* three uninterpreted bytes */
Datum Data[9];        /* 3 consecutive 3 byte vectors */
```

Variable-length vectors are defined by specifying a subrange of legal lengths, inclusively, using the notation <floor..ceiling>. When





these are encoded, the actual length precedes the vector's contents in the byte stream. The length will be in the form of a number consuming as many bytes as required to hold the vector's specified maximum (ceiling) length. A variable-length vector with an actual length field of zero is referred to as an empty vector.

```
T T'<floor..ceiling>;
```

In the following example, mandatory is a vector that must contain between 300 and 400 bytes of type opaque. It can never be empty. The actual length field consumes two bytes, a uint16, sufficient to represent the value 400 (see Section 4.4). On the other hand, longer can represent up to 800 bytes of data, or 400 uint16 elements, and it may be empty. Its encoding will include a two-byte actual length field prepended to the vector. The length of an encoded vector must be an even multiple of the length of a single element (for example, a 17-byte vector of uint16 would be illegal).

```
opaque mandatory<300..400>;
      /* length field is 2 bytes, cannot be empty */
uint16 longer<0..800>;
      /* zero to 400 16-bit unsigned integers */
```

### 4.4. Numbers

The basic numeric data type is an unsigned byte (uint8). All larger numeric data types are formed from fixed-length series of bytes concatenated as described in Section 4.1 and are also unsigned. The following numeric types are predefined.

```
uint8 uint16[2];
uint8 uint24[3];
uint8 uint32[4];
uint8 uint64[8];
```

All values, here and elsewhere in the specification, are stored in "network" or "big-endian" order; the uint32 represented by the hex bytes 01 02 03 04 is equivalent to the decimal value 16909060.

### 4.5. Enumerateds

An additional sparse data type is available called enum. A field of

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 963
Keysight, EX-1045
Page 963

type enum can only assume the values declared in the definition. Each definition is a different type. Only enumerateds of the same type may be assigned or compared. Every element of an enumerated must be assigned a value, as demonstrated in the following example. Since the elements of the enumerated are not ordered, they can be assigned any unique value, in any order.



```
enum { e1(v1), e2(v2), ... , en(vn) [[, (n)]] } Te;
```

Enumerateds occupy as much space in the byte stream as would its maximal defined ordinal value. The following definition would cause one byte to be used to carry fields of type Color.

```
enum { red(3), blue(5), white(7) } Color;
```

One may optionally specify a value without its associated tag to force the width definition without defining a superfluous element. In the following example, Taste will consume two bytes in the data stream but can only assume the values 1, 2, or 4.

```
enum { sweet(1), sour(2), bitter(4), (32000) } Taste;
```

The names of the elements of an enumeration are scoped within the defined type. In the first example, a fully qualified reference to the second element of the enumeration would be Color.blue. Such qualification is not required if the target of the assignment is well specified.

```
Color color = Color.blue;     /* overspecified, legal */
Color color = blue;           /* correct, type implicit */
```

For enumerateds that are never converted to external representation, the numerical information may be omitted.

```
enum { low, medium, high } Amount;
```

### 4.6. Constructed Types

Structure types may be constructed from primitive types for convenience. Each specification declares a new, unique type. The syntax for definition is much like that of C.

```
struct {
  T1 f1;
  T2 f2;
  ...
  Tn fn;
} [[T]];
```

The fields within a structure may be qualified using the type's name, with a syntax much like that available for enumerateds. For example, T.f2 refers to the second field of the previous declaration. Structure definitions may be embedded.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

### 4.6.1. Variants

Defined structures may have variants based on some knowledge that is available within the environment. The selector must be an enumerated type that defines the possible variants the structure defines. There must be a case arm for every element of the enumeration declared in the select. The body of the variant structure may be given a label for reference. The mechanism by which the variant is selected at runtime is not prescribed by the presentation language.

```
struct {
    T1 f1;
    T2 f2;
    ....
    Tn fn;
    select (E) {
        case e1: Te1;
        case e2: Te2;
        ....
        case en: Ten;
    } [[fv]];
} [[Tv]];
```

For example:

```
enum { apple, orange } VariantTag;
struct {
    uint16 number;
    opaque string<0..10>; /* variable length */
} V1;
struct {
    uint32 number;
    opaque string[10];    /* fixed length */
} V2;
struct {
    select (VariantTag) { /* value of selector is implicit */
        case apple: V1;   /* VariantBody, tag = apple */
        case orange: V2;  /* VariantBody, tag = orange */
    } variant_body;       /* optional label on variant */
} VariantRecord;
```

Variant structures may be qualified (narrowed) by specifying a value for the selector prior to the type. For example, an

```
orange VariantRecord
```

is a narrowed type of a VariantRecord containing a variant_body of type V2.



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

### 4.7. Cryptographic Attributes

The four cryptographic operations digital signing, stream cipher encryption, block cipher encryption, and public key encryption are designated digitally-signed, stream-ciphered, block-ciphered, and public-key-encrypted, respectively. A field's cryptographic processing is specified by prepending an appropriate key word designation before the field's type specification. Cryptographic keys are implied by the current session state (see Section 6.1).

In digital signing, one-way hash functions are used as input for a signing algorithm. A digitally-signed element is encoded as an opaque vector <0..2^16-1>, where the length is specified by the signing algorithm and key.

In RSA signing, a 36-byte structure of two hashes (one SHA and one MD5) is signed (encrypted with the private key). It is encoded with PKCS #1 block type 1, as described in [PKCS1A].

Note: The standard reference for PKCS#1 is now RFC 3447 [PKCS1B]. However, to minimize differences with TLS 1.0 text, we are using the terminology of RFC 2313 [PKCS1A].

In DSS, the 20 bytes of the SHA hash are run directly through the Digital Signing Algorithm with no additional hashing. This produces two values, r and s. The DSS signature is an opaque vector, as above, the contents of which are the DER encoding of:

```
Dss-Sig-Value  ::=  SEQUENCE  {
     r       INTEGER,
     s       INTEGER
}
```

In stream cipher encryption, the plaintext is exclusive-ORed with an identical amount of output generated from a cryptographically secure keyed pseudorandom number generator.

In block cipher encryption, every block of plaintext encrypts to a block of ciphertext. All block cipher encryption is done in CBC (Cipher Block Chaining) mode, and all items that are block-ciphered will be an exact multiple of the cipher block length.

In public key encryption, a public key algorithm is used to encrypt data in such a way that it can be decrypted only with the matching private key. A public-key-encrypted element is encoded as an opaque vector <0..2^16-1>, where the length is specified by the signing algorithm and key.





An RSA-encrypted value is encoded with PKCS #1 block type 2, as described in [PKCS1A].

In the following example,

```
stream-ciphered struct {
    uint8 field1;
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 966
Keysight, EX-1045
Page 966

```
        uint8 field2;
        digitally-signed opaque hash[20];
    } UserType;
```

the contents of hash are used as input for the signing algorithm, and then the entire structure is encrypted with a stream cipher. The length of this structure, in bytes, would be equal to two bytes for field1 and field2, plus two bytes for the length of the signature, plus the length of the output of the signing algorithm. This is known because the algorithm and key used for the signing are known prior to encoding or decoding this structure.

### 4.8. Constants

Typed constants can be defined for purposes of specification by declaring a symbol of the desired type and assigning values to it. Under-specified types (opaque, variable length vectors, and structures that contain opaque) cannot be assigned values. No fields of a multi-element structure or vector may be elided.

For example:

```
    struct {
        uint8 f1;
        uint8 f2;
    } Example1;

    Example1 ex1 = {1, 4};  /* assigns f1 = 1, f2 = 4 */
```

## 5. HMAC and the Pseudorandom Function

A number of operations in the TLS record and handshake layer require a keyed MAC; this is a secure digest of some data protected by a secret. Forging the MAC is infeasible without knowledge of the MAC secret. The construction we use for this operation is known as HMAC, and is described in [HMAC].

HMAC can be used with a variety of different hash algorithms. TLS uses it in the handshake with two different algorithms, MD5 and SHA-1, denoting these as HMAC_MD5(secret, data) and HMAC_SHA(secret, data). Additional hash algorithms can be defined by cipher suites





and used to protect record data, but MD5 and SHA-1 are hard coded into the description of the handshaking for this version of the protocol.

In addition, a construction is required to do expansion of secrets into blocks of data for the purposes of key generation or validation. This pseudo-random function (PRF) takes as input a secret, a seed, and an identifying label and produces an output of arbitrary length.

In order to make the PRF as secure as possible, it uses two hash algorithms in a way that should guarantee its security if either algorithm remains secure.

First, we define a data expansion function, P_hash(secret, data) that

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 967
Keysight, EX-1045
Page 967

uses a single hash function to expand a secret and seed into an arbitrary quantity of output:

```
P_hash(secret, seed) = HMAC_hash(secret, A(1) + seed) +
                       HMAC_hash(secret, A(2) + seed) +
                       HMAC_hash(secret, A(3) + seed) + ...
```

Where + indicates concatenation.

A() is defined as:

```
A(0) = seed
A(i) = HMAC_hash(secret, A(i-1))
```

P_hash can be iterated as many times as is necessary to produce the required quantity of data. For example, if P_SHA-1 is being used to create 64 bytes of data, it will have to be iterated 4 times (through A(4)), creating 80 bytes of output data; the last 16 bytes of the final iteration will then be discarded, leaving 64 bytes of output data.

TLS's PRF is created by splitting the secret into two halves and using one half to generate data with P_MD5 and the other half to generate data with P_SHA-1, then exclusive-ORing the outputs of these two expansion functions together.

S1 and S2 are the two halves of the secret, and each is the same length. S1 is taken from the first half of the secret, S2 from the second half. Their length is created by rounding up the length of the overall secret, divided by two; thus, if the original secret is an odd number of bytes long, the last byte of S1 will be the same as the first byte of S2.





```
L_S = length in bytes of secret;
L_S1 = L_S2 = ceil(L_S / 2);
```

The secret is partitioned into two halves (with the possibility of one shared byte) as described above, S1 taking the first L_S1 bytes, and S2 the last L_S2 bytes.

The PRF is then defined as the result of mixing the two pseudorandom streams by exclusive-ORing them together.

```
PRF(secret, label, seed) = P_MD5(S1, label + seed) XOR
                           P_SHA-1(S2, label + seed);
```

The label is an ASCII string. It should be included in the exact form it is given without a length byte or trailing null character. For example, the label "slithy toves" would be processed by hashing the following bytes:

```
73 6C 69 74 68 79 20 74 6F 76 65 73
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

Note that because MD5 produces 16-byte outputs and SHA-1 produces 20-byte outputs, the boundaries of their internal iterations will not be aligned. Generating an 80-byte output will require that P_MD5 iterate through A(5), while P_SHA-1 will only iterate through A(4).

## 6. The TLS Record Protocol

The TLS Record Protocol is a layered protocol. At each layer, messages may include fields for length, description, and content. The Record Protocol takes messages to be transmitted, fragments the data into manageable blocks, optionally compresses the data, applies a MAC, encrypts, and transmits the result. Received data is decrypted, verified, decompressed, reassembled, and then delivered to higher-level clients.

Four record protocol clients are described in this document: the handshake protocol, the alert protocol, the change cipher spec protocol, and the application data protocol. In order to allow extension of the TLS protocol, additional record types can be supported by the record protocol. Any new record types SHOULD allocate type values immediately beyond the ContentType values for the four record types described here (see Appendix A.1). All such values must be defined by RFC 2434 Standards Action. See Section 11 for IANA Considerations for ContentType values.

If a TLS implementation receives a record type it does not understand, it SHOULD just ignore it. Any protocol designed for use





over TLS MUST be carefully designed to deal with all possible attacks against it. Note that because the type and length of a record are not protected by encryption, care SHOULD be taken to minimize the value of traffic analysis of these values.

### 6.1. Connection States

A TLS connection state is the operating environment of the TLS Record Protocol. It specifies a compression algorithm, and encryption algorithm, and a MAC algorithm. In addition, the parameters for these algorithms are known: the MAC secret and the bulk encryption keys for the connection in both the read and the write directions. Logically, there are always four connection states outstanding: the current read and write states, and the pending read and write states. All records are processed under the current read and write states. The security parameters for the pending states can be set by the TLS Handshake Protocol, and the Change Cipher Spec can selectively make either of the pending states current, in which case the appropriate current state is disposed of and replaced with the pending state; the pending state is then reinitialized to an empty state. It is illegal to make a state that has not been initialized with security parameters a current state. The initial current state always specifies that no encryption, compression, or MAC will be used.

The security parameters for a TLS Connection read and write state are set by providing the following values:

connection end

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 969
Keysight, EX-1045
Page 969

Whether this entity is considered the "client" or the "server" in this connection.

bulk encryption algorithm
An algorithm to be used for bulk encryption. This specification includes the key size of this algorithm, how much of that key is secret, whether it is a block or stream cipher, and the block size of the cipher (if appropriate).

MAC algorithm
An algorithm to be used for message authentication. This specification includes the size of the hash returned by the MAC algorithm.

compression algorithm
An algorithm to be used for data compression. This specification must include all information the algorithm requires compression.

master secret
A 48-byte secret shared between the two peers in the connection.





client random
A 32-byte value provided by the client.

server random
A 32-byte value provided by the server.

These parameters are defined in the presentation language as:

```
enum { server, client } ConnectionEnd;

enum { null, rc4, rc2, des, 3des, des40, idea, aes }
BulkCipherAlgorithm;

enum { stream, block } CipherType;

enum { null, md5, sha } MACAlgorithm;

enum { null(0), (255) } CompressionMethod;

/* The algorithms specified in CompressionMethod,
   BulkCipherAlgorithm, and MACAlgorithm may be added to. */

struct {
    ConnectionEnd          entity;
    BulkCipherAlgorithm    bulk_cipher_algorithm;
    CipherType             cipher_type;
    uint8                  key_size;
    uint8                  key_material_length;
    MACAlgorithm           mac_algorithm;
    uint8                  hash_size;
    CompressionMethod      compression_algorithm;
    opaque                 master_secret[48];
    opaque                 client_random[32];
    opaque                 server_random[32];
} SecurityParameters;
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 970
Keysight, EX-1045
Page 970

The record layer will use the security parameters to generate the following four items:

```
client write MAC secret
server write MAC secret
client write key
server write key
```

The client write parameters are used by the server when receiving and processing records and vice-versa. The algorithm used for generating these items from the security parameters is described in Section 6.3.





Once the security parameters have been set and the keys have been generated, the connection states can be instantiated by making them the current states. These current states MUST be updated for each record processed. Each connection state includes the following elements:

compression state
: The current state of the compression algorithm.

cipher state
: The current state of the encryption algorithm. This will consist of the scheduled key for that connection. For stream ciphers, this will also contain whatever state information is necessary to allow the stream to continue to encrypt or decrypt data.

MAC secret
: The MAC secret for this connection, as generated above.

sequence number
: Each connection state contains a sequence number, which is maintained separately for read and write states. The sequence number MUST be set to zero whenever a connection state is made the active state. Sequence numbers are of type uint64 and may not exceed 2^64-1. Sequence numbers do not wrap. If a TLS implementation would need to wrap a sequence number, it must renegotiate instead. A sequence number is incremented after each record: specifically, the first record transmitted under a particular connection state MUST use sequence number 0.

## 6.2. Record layer

The TLS Record Layer receives uninterpreted data from higher layers in non-empty blocks of arbitrary size.

### 6.2.1. Fragmentation

The record layer fragments information blocks into TLSPlaintext records carrying data in chunks of 2^14 bytes or less. Client message boundaries are not preserved in the record layer (i.e., multiple client messages of the same ContentType MAY be coalesced into a single TLSPlaintext record, or a single message MAY be fragmented across several records).

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

```
       struct {
           uint8 major, minor;
       } ProtocolVersion;

       enum {
           change_cipher_spec(20), alert(21), handshake(22),
           application_data(23), (255)
       } ContentType;

       struct {
           ContentType type;
           ProtocolVersion version;
           uint16 length;
           opaque fragment[TLSPlaintext.length];
       } TLSPlaintext;
```

type
The higher-level protocol used to process the enclosed fragment.

version
The version of the protocol being employed. This document describes TLS Version 1.1, which uses the version { 3, 2 }. The version value 3.2 is historical: TLS version 1.1 is a minor modification to the TLS 1.0 protocol, which was itself a minor modification to the SSL 3.0 protocol, which bears the version value 3.0. (See Appendix A.1.)

length
The length (in bytes) of the following TLSPlaintext.fragment. The length should not exceed 2^14.

fragment
The application data. This data is transparent and is treated as an independent block to be dealt with by the higher-level protocol specified by the type field.

Note: Data of different TLS Record layer content types MAY be interleaved. Application data is generally of lower precedence for transmission than other content types. However, records MUST be delivered to the network in the same order as they are protected by the record layer. Recipients MUST receive and process interleaved application layer traffic during handshakes subsequent to the first one on a connection.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

### 6.2.2. Record Compression and Decompression

All records are compressed using the compression algorithm defined in the current session state. There is always an active compression algorithm; however, initially it is defined as CompressionMethod.null. The compression algorithm translates a TLSPlaintext structure into a TLSCompressed structure. Compression functions are initialized with default state information whenever a connection state is made active.

Compression must be lossless and may not increase the content length by more than 1024 bytes. If the decompression function encounters a TLSCompressed.fragment that would decompress to a length in excess of 2^14 bytes, it should report a fatal decompression failure error.

```
    struct {
        ContentType type;       /* same as TLSPlaintext.type */
        ProtocolVersion version;/* same as TLSPlaintext.version */
        uint16 length;
        opaque fragment[TLSCompressed.length];
    } TLSCompressed;
```

length
The length (in bytes) of the following TLSCompressed.fragment. The length should not exceed 2^14 + 1024.

fragment
The compressed form of TLSPlaintext.fragment.

Note: A CompressionMethod.null operation is an identity operation; no fields are altered.

Implementation note: Decompression functions are responsible for ensuring that messages cannot cause internal buffer overflows.

### 6.2.3. Record Payload Protection

The encryption and MAC functions translate a TLSCompressed structure into a TLSCiphertext. The decryption functions reverse the process. The MAC of the record also includes a sequence number so that missing, extra, or repeated messages are detectable.



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

```
        struct {
            ContentType type;
            ProtocolVersion version;
            uint16 length;
            select (CipherSpec.cipher_type) {
                case stream: GenericStreamCipher;
                case block: GenericBlockCipher;
            } fragment;
        } TLSCiphertext;
```

type
   The type field is identical to TLSCompressed.type.

version
   The version field is identical to TLSCompressed.version.

length
   The length (in bytes) of the following TLSCiphertext.fragment.
   The length may not exceed 2^14 + 2048.

fragment
   The encrypted form of TLSCompressed.fragment, with the MAC.

#### 6.2.3.1. Null or Standard Stream Cipher

Stream ciphers (including BulkCipherAlgorithm.null, see Appendix A.6) convert TLSCompressed.fragment structures to and from stream TLSCiphertext.fragment structures.

```
        stream-ciphered struct {
            opaque content[TLSCompressed.length];
            opaque MAC[CipherSpec.hash_size];
        } GenericStreamCipher;
```

The MAC is generated as:

```
        HMAC_hash(MAC_write_secret, seq_num + TLSCompressed.type +
                      TLSCompressed.version + TLSCompressed.length +
                      TLSCompressed.fragment));
```

where "+" denotes concatenation.

seq_num
   The sequence number for this record.

hash
   The hashing algorithm specified by
   SecurityParameters.mac_algorithm.





Note that the MAC is computed before encryption. The stream cipher encrypts the entire block, including the MAC. For stream ciphers that do not use a synchronization vector (such as RC4), the stream cipher state from the end of one record is simply used on the subsequent packet. If the CipherSuite is TLS_NULL_WITH_NULL_NULL, encryption consists of the identity operation (i.e., the data is not encrypted, and the MAC size is zero, implying that no MAC is used).

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

TLSCiphertext.length is TLSCompressed.length plus
CipherSpec.hash_size.

6.2.3.2. CBC Block Cipher

For block ciphers (such as RC2, DES, or AES), the encryption and MAC functions convert TLSCompressed.fragment structures to and from block TLSCiphertext.fragment structures.

```
block-ciphered struct {
    opaque IV[CipherSpec.block_length];
    opaque content[TLSCompressed.length];
    opaque MAC[CipherSpec.hash_size];
    uint8 padding[GenericBlockCipher.padding_length];
    uint8 padding_length;
} GenericBlockCipher;
```

The MAC is generated as described in Section 6.2.3.1.

IV

Unlike previous versions of SSL and TLS, TLS 1.1 uses an explicit IV in order to prevent the attacks described by [CBCATT]. We recommend the following equivalently strong procedures. For clarity we use the following notation.

IV
The transmitted value of the IV field in the GenericBlockCipher structure.

CBC residue
The last ciphertext block of the previous record.

mask
The actual value that the cipher XORs with the plaintext prior to encryption of the first cipher block of the record.

In prior versions of TLS, there was no IV field and the CBC residue and mask were one and the same. See Sections 6.1, 6.2.3.2, and 6.3, of [TLS1.0] for details of TLS 1.0 IV handling.





One of the following two algorithms SHOULD be used to generate the per-record IV:

(1) Generate a cryptographically strong random string R of length CipherSpec.block_length. Place R in the IV field. Set the mask to R. Thus, the first cipher block will be encrypted as E(R XOR Data).

(2) Generate a cryptographically strong random number R of length CipherSpec.block_length and prepend it to the plaintext prior to encryption. In this case either:

(a) The cipher may use a fixed mask such as zero.
(b) The CBC residue from the previous record may be used as

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

the mask. This preserves maximum code compatibility with TLS 1.0 and SSL 3. It also has the advantage that it does not require the ability to quickly reset the IV, which is known to be a problem on some systems.

In either (2)(a) or (2)(b) the data (R || data) is fed into the encryption process. The first cipher block (containing E(mask XOR R) is placed in the IV field. The first block of content contains E(IV XOR data).

The following alternative procedure MAY be used; however, it has not been demonstrated to be as cryptographically strong as the above procedures. The sender prepends a fixed block F to the plaintext (or, alternatively, a block generated with a weak PRNG). He then encrypts as in (2), above, using the CBC residue from the previous block as the mask for the prepended block. Note that in this case the mask for the first record transmitted by the application (the Finished) MUST be generated using a cryptographically strong PRNG.

The decryption operation for all three alternatives is the same. The receiver decrypts the entire GenericBlockCipher structure and then discards the first cipher block, corresponding to the IV component.

padding
Padding that is added to force the length of the plaintext to be an integral multiple of the block cipher's block length. The padding MAY be any length up to 255 bytes, as long as it results in the TLSCiphertext.length being an integral multiple of the block length. Lengths longer than necessary might be desirable to frustrate attacks on a protocol that are based on analysis of the lengths of exchanged messages. Each uint8 in the padding data vector MUST be filled with the padding length value. The receiver





MUST check this padding and SHOULD use the bad_record_mac alert to indicate padding errors.

padding_length
The padding length MUST be such that the total size of the GenericBlockCipher structure is a multiple of the cipher's block length. Legal values range from zero to 255, inclusive. This length specifies the length of the padding field exclusive of the padding_length field itself.

The encrypted data length (TLSCiphertext.length) is one more than the sum of CipherSpec.block_length, TLSCompressed.length, CipherSpec.hash_size, and padding_length.

Example: If the block length is 8 bytes, the content length (TLSCompressed.length) is 61 bytes, and the MAC length is 20 bytes, then the length before padding is 82 bytes (this does not include the IV, which may or may not be encrypted, as discussed above). Thus, the padding length modulo 8 must be equal to 6 in order to make the total length an even multiple of 8 bytes (the block length). The padding length

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

can be 6, 14, 22, and so on, through 254. If the padding length were the minimum necessary, 6, the padding would be 6 bytes, each containing the value 6. Thus, the last 8 octets of the GenericBlockCipher before block encryption would be xx 06 06 06 06 06 06 06, where xx is the last octet of the MAC.

Note: With block ciphers in CBC mode (Cipher Block Chaining), it is critical that the entire plaintext of the record be known before any ciphertext is transmitted. Otherwise, it is possible for the attacker to mount the attack described in [CBCATT].

Implementation Note: Canvel et al. [CBCTIME] have demonstrated a timing attack on CBC padding based on the time required to compute the MAC. In order to defend against this attack, implementations MUST ensure that record processing time is essentially the same whether or not the padding is correct. In general, the best way to do this is to compute the MAC even if the padding is incorrect, and only then reject the packet. For instance, if the pad appears to be incorrect, the implementation might assume a zero-length pad and then compute the MAC. This leaves a small timing channel, since MAC performance depends to some extent on the size of the data fragment,





but it is not believed to be large enough to be exploitable, due to the large block size of existing MACs and the small size of the timing signal.

### 6.3. Key Calculation

The Record Protocol requires an algorithm to generate keys, and MAC secrets from the security parameters provided by the handshake protocol.

The master secret is hashed into a sequence of secure bytes, which are assigned to the MAC secrets and keys required by the current connection state (see Appendix A.6). CipherSpecs require a client write MAC secret, a server write MAC secret, a client write key, and a server write key, each of which is generated from the master secret in that order. Unused values are empty.

When keys and MAC secrets are generated, the master secret is used as an entropy source.

To generate the key material, compute

```
key_block = PRF(SecurityParameters.master_secret,
                "key expansion",
                SecurityParameters.server_random +
          SecurityParameters.client_random);
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 977
Keysight, EX-1045
Page 977

until enough output has been generated. Then the key_block is partitioned as follows:

```
client_write_MAC_secret[SecurityParameters.hash_size]
server_write_MAC_secret[SecurityParameters.hash_size]
client_write_key[SecurityParameters.key_material_length]
server_write_key[SecurityParameters.key_material_length]
```

Implementation note: The currently defined cipher suite that requires the most material is AES_256_CBC_SHA, defined in [TLSAES]. It requires 2 x 32 byte keys, 2 x 20 byte MAC secrets, and 2 x 16 byte Initialization Vectors, for a total of 136 bytes of key material.

## 7. The TLS Handshaking Protocols

TLS has three subprotocols that are used to allow peers to agree upon security parameters for the record layer, to authenticate themselves, to instantiate negotiated security parameters, and to report error conditions to each other.





The Handshake Protocol is responsible for negotiating a session, which consists of the following items:

session identifier
: An arbitrary byte sequence chosen by the server to identify an active or resumable session state.

peer certificate
: X509v3 [X509] certificate of the peer. This element of the state may be null.

compression method
: The algorithm used to compress data prior to encryption.

cipher spec
: Specifies the bulk data encryption algorithm (such as null, DES, etc.) and a MAC algorithm (such as MD5 or SHA). It also defines cryptographic attributes such as the hash_size. (See Appendix A.6 for formal definition.)

master secret
: 48-byte secret shared between the client and server.

is resumable
: A flag indicating whether the session can be used to initiate new connections.

These items are then used to create security parameters for use by the Record Layer when protecting application data. Many connections can be instantiated using the same session through the resumption feature of the TLS Handshake Protocol.

### 7.1. Change Cipher Spec Protocol

The change cipher spec protocol exists to signal transitions in

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

ciphering strategies. The protocol consists of a single message, which is encrypted and compressed under the current (not the pending) connection state. The message consists of a single byte of value 1.

```
struct {
    enum { change_cipher_spec(1), (255) } type;
} ChangeCipherSpec;
```

The change cipher spec message is sent by both the client and the server to notify the receiving party that subsequent records will be protected under the newly negotiated CipherSpec and keys. Reception of this message causes the receiver to instruct the Record Layer to immediately copy the read pending state into the read current state.





Immediately after sending this message, the sender MUST instruct the record layer to make the write pending state the write active state. (See Section 6.1.) The change cipher spec message is sent during the handshake after the security parameters have been agreed upon, but before the verifying finished message is sent (see Section 7.4.9).

Note: If a rehandshake occurs while data is flowing on a connection, the communicating parties may continue to send data using the old CipherSpec. However, once the ChangeCipherSpec has been sent, the new CipherSpec MUST be used. The first side to send the ChangeCipherSpec does not know that the other side has finished computing the new keying material (e.g., if it has to perform a time consuming public key operation). Thus, a small window of time, during which the recipient must buffer the data, MAY exist. In practice, with modern machines this interval is likely to be fairly short.

## 7.2. Alert Protocol

One of the content types supported by the TLS Record layer is the alert type. Alert messages convey the severity of the message and a description of the alert. Alert messages with a level of fatal result in the immediate termination of the connection. In this case, other connections corresponding to the session may continue, but the session identifier MUST be invalidated, preventing the failed session from being used to establish new connections. Like other messages, alert messages are encrypted and compressed, as specified by the current connection state.

```
enum { warning(1), fatal(2), (255) } AlertLevel;

enum {
    close_notify(0),
    unexpected_message(10),
    bad_record_mac(20),
    decryption_failed(21),
    record_overflow(22),
    decompression_failure(30),
    handshake_failure(40),
    no_certificate_RESERVED (41),
    bad_certificate(42),
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 979
Keysight, EX-1045
Page 979

```
                unsupported_certificate(43),
                certificate_revoked(44),
                certificate_expired(45),
                certificate_unknown(46),
                illegal_parameter(47),
                unknown_ca(48),
```



```
                access_denied(49),
                decode_error(50),
                decrypt_error(51),
                export_restriction_RESERVED(60),
                protocol_version(70),
                insufficient_security(71),
                internal_error(80),
                user_canceled(90),
                no_renegotiation(100),
                (255)
            } AlertDescription;

            struct {
                AlertLevel level;
                AlertDescription description;
            } Alert;
```

### 7.2.1. Closure Alerts

The client and the server must share knowledge that the connection is ending in order to avoid a truncation attack. Either party may initiate the exchange of closing messages.

close_notify
: This message notifies the recipient that the sender will not send any more messages on this connection. Note that as of TLS 1.1, failure to properly close a connection no longer requires that a session not be resumed. This is a change from TLS 1.0 to conform with widespread implementation practice.

Either party may initiate a close by sending a close_notify alert. Any data received after a closure alert is ignored.

Unless some other fatal alert has been transmitted, each party is required to send a close_notify alert before closing the write side of the connection. The other party MUST respond with a close_notify alert of its own and close down the connection immediately, discarding any pending writes. It is not required for the initiator of the close to wait for the responding close_notify alert before closing the read side of the connection.

If the application protocol using TLS provides that any data may be carried over the underlying transport after the TLS connection is closed, the TLS implementation must receive the responding close_notify alert before indicating to the application layer that the TLS connection has ended. If the application protocol will not transfer any additional data, but will only close the underlying transport connection, then the implementation MAY choose to close the

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

transport without waiting for the responding close_notify. No part of this standard should be taken to dictate the manner in which a usage profile for TLS manages its data transport, including when connections are opened or closed.

Note: It is assumed that closing a connection reliably delivers pending data before destroying the transport.

7.2.2. Error Alerts

Error handling in the TLS Handshake protocol is very simple. When an error is detected, the detecting party sends a message to the other party. Upon transmission or receipt of a fatal alert message, both parties immediately close the connection. Servers and clients MUST forget any session-identifiers, keys, and secrets associated with a failed connection. Thus, any connection terminated with a fatal alert MUST NOT be resumed. The following error alerts are defined:

unexpected_message
An inappropriate message was received. This alert is always fatal and should never be observed in communication between proper implementations.

bad_record_mac
This alert is returned if a record is received with an incorrect MAC. This alert also MUST be returned if an alert is sent because a TLSCiphertext decrypted in an invalid way: either it wasn't an even multiple of the block length, or its padding values, when checked, weren't correct. This message is always fatal.

decryption_failed
This alert MAY be returned if a TLSCiphertext decrypted in an invalid way: either it wasn't an even multiple of the block length, or its padding values, when checked, weren't correct. This message is always fatal.

Note: Differentiating between bad_record_mac and decryption_failed alerts may permit certain attacks against CBC mode as used in TLS [CBCATT]. It is preferable to uniformly use the bad_record_mac alert to hide the specific type of the error.

record_overflow
A TLSCiphertext record was received that had a length more than 2^14+2048 bytes, or a record decrypted to a TLSCompressed record with more than 2^14+1024 bytes. This message is always fatal.



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

decompression_failure
      The decompression function received improper input (e.g., data
      that would expand to excessive length).  This message is always
      fatal.

handshake_failure
      Reception of a handshake_failure alert message indicates that
      the sender was unable to negotiate an acceptable set of
      security parameters given the options available.  This is a
      fatal error.

no_certificate_RESERVED
      This alert was used in SSLv3 but not in TLS.  It should not be
      sent by compliant implementations.

bad_certificate
      A certificate was corrupt, contained signatures that did not
      verify correctly, etc.

unsupported_certificate
      A certificate was of an unsupported type.

certificate_revoked
      A certificate was revoked by its signer.

certificate_expired
      A certificate has expired or is not currently valid.

certificate_unknown
      Some other (unspecified) issue arose in processing the
      certificate, rendering it unacceptable.

illegal_parameter
      A field in the handshake was out of range or inconsistent with
      other fields.  This is always fatal.

unknown_ca
      A valid certificate chain or partial chain was received, but
      the certificate was not accepted because the CA certificate
      could not be located or couldn't be matched with a known,
      trusted CA.  This message is always fatal.

access_denied
      A valid certificate was received, but when access control was
      applied, the sender decided not to proceed with negotiation.
      This message is always fatal.





decode_error
      A message could not be decoded because some field was out of
      the specified range or the length of the message was incorrect.
      This message is always fatal.

decrypt_error
      A handshake cryptographic operation failed, including being

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 982
Keysight, EX-1045
Page 982

unable to correctly verify a signature, decrypt a key exchange, or validate a finished message.

export_restriction_RESERVED
This alert was used in TLS 1.0 but not TLS 1.1.

protocol_version
The protocol version the client has attempted to negotiate is recognized but not supported. (For example, old protocol versions might be avoided for security reasons). This message is always fatal.

insufficient_security
Returned instead of handshake_failure when a negotiation has failed specifically because the server requires ciphers more secure than those supported by the client. This message is always fatal.

internal_error
An internal error unrelated to the peer or the correctness of the protocol (such as a memory allocation failure) makes it impossible to continue. This message is always fatal.

user_canceled
This handshake is being canceled for some reason unrelated to a protocol failure. If the user cancels an operation after the handshake is complete, just closing the connection by sending a close_notify is more appropriate. This alert should be followed by a close_notify. This message is generally a warning.

no_renegotiation
Sent by the client in response to a hello request or by the server in response to a client hello after initial handshaking. Either of these would normally lead to renegotiation; when that is not appropriate, the recipient should respond with this alert. At that point, the original requester can decide whether to proceed with the connection. One case where this would be appropriate is where a server has spawned a process to satisfy a request; the process might receive security parameters (key length, authentication, etc.) at startup and it




might be difficult to communicate changes to these parameters after that point. This message is always a warning.

For all errors where an alert level is not explicitly specified, the sending party MAY determine at its discretion whether this is a fatal error or not; if an alert with a level of warning is received, the receiving party MAY decide at its discretion whether to treat this as a fatal error or not. However, all messages that are transmitted with a level of fatal MUST be treated as fatal messages.

New alert values MUST be defined by RFC 2434 Standards Action. See Section 11 for IANA Considerations for alert values.

### 7.3. Handshake Protocol Overview

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 983
Keysight, EX-1045
Page 983

The cryptographic parameters of the session state are produced by the TLS Handshake Protocol, which operates on top of the TLS Record Layer. When a TLS client and server first start communicating, they agree on a protocol version, select cryptographic algorithms, optionally authenticate each other, and use public-key encryption techniques to generate shared secrets.

The TLS Handshake Protocol involves the following steps:

- Exchange hello messages to agree on algorithms, exchange random values, and check for session resumption.

- Exchange the necessary cryptographic parameters to allow the client and server to agree on a premaster secret.

- Exchange certificates and cryptographic information to allow the client and server to authenticate themselves.

- Generate a master secret from the premaster secret and exchanged random values.

- Provide security parameters to the record layer.

- Allow the client and server to verify that their peer has calculated the same security parameters and that the handshake occurred without tampering by an attacker.

Note that higher layers should not be overly reliant on whether TLS always negotiates the strongest possible connection between two peers. There are a number of ways in which a man-in-the-middle attacker can attempt to make two entities drop down to the least secure method they support. The protocol has been designed to minimize this risk, but there are still attacks available. For





example, an attacker could block access to the port a secure service runs on, or attempt to get the peers to negotiate an unauthenticated connection. The fundamental rule is that higher levels must be cognizant of what their security requirements are and never transmit information over a channel less secure than what they require. The TLS protocol is secure in that any cipher suite offers its promised level of security: if you negotiate 3DES with a 1024 bit RSA key exchange with a host whose certificate you have verified, you can expect to be that secure.

However, one SHOULD never send data over a link encrypted with 40-bit security unless one feels that data is worth no more than the effort required to break that encryption.

These goals are achieved by the handshake protocol, which can be summarized as follows: The client sends a client hello message to which the server must respond with a server hello message, or else a fatal error will occur and the connection will fail. The client hello and server hello are used to establish security enhancement capabilities between client and server. The client hello and server hello establish the following attributes: Protocol Version, Session

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

ID, Cipher Suite, and Compression Method. Additionally, two random values are generated and exchanged: ClientHello.random and ServerHello.random.

The actual key exchange uses up to four messages: the server certificate, the server key exchange, the client certificate, and the client key exchange. New key exchange methods can be created by specifying a format for these messages and by defining the use of the messages to allow the client and server to agree upon a shared secret. This secret MUST be quite long; currently defined key exchange methods exchange secrets that range from 48 to 128 bytes in length.

Following the hello messages, the server will send its certificate, if it is to be authenticated. Additionally, a server key exchange message may be sent, if it is required (e.g., if the server has no certificate, or if its certificate is for signing only). If the server is authenticated, it may request a certificate from the client, if that is appropriate to the cipher suite selected. Next, the server will send the server hello done message, indicating that the hello-message phase of the handshake is complete. The server will then wait for a client response. If the server has sent a certificate request message, the client must send the certificate message. The client key exchange message is now sent, and the content of that message will depend on the public key algorithm selected between the client hello and the server hello. If the client has sent a certificate with signing ability, a digitally-





signed certificate verify message is sent to explicitly verify the certificate.

At this point, a change cipher spec message is sent by the client, and the client copies the pending Cipher Spec into the current Cipher Spec. The client then immediately sends the finished message under the new algorithms, keys, and secrets. In response, the server will send its own change cipher spec message, transfer the pending to the current Cipher Spec, and send its finished message under the new Cipher Spec. At this point, the handshake is complete, and the client and server may begin to exchange application layer data. (See flow chart below.) Application data MUST NOT be sent prior to the completion of the first handshake (before a cipher suite other TLS_NULL_WITH_NULL_NULL is established).

```
   Client                                               Server

   ClientHello                  -------->
                                                   ServerHello
                                                  Certificate*
                                            ServerKeyExchange*
                                           CertificateRequest*
                                <--------      ServerHelloDone
   Certificate*
   ClientKeyExchange
   CertificateVerify*
   [ChangeCipherSpec]
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 985
Keysight, EX-1045
Page 985

```
   Finished                    -------->
                                           [ChangeCipherSpec]
                               <--------             Finished
   Application Data            <-------->     Application Data
```

Fig. 1. Message flow for a full handshake

* Indicates optional or situation-dependent messages that are not always sent.

Note: To help avoid pipeline stalls, ChangeCipherSpec is an independent TLS Protocol content type, and is not actually a TLS handshake message.

When the client and server decide to resume a previous session or duplicate an existing session (instead of negotiating new security parameters), the message flow is as follows:

The client sends a ClientHello using the Session ID of the session to be resumed. The server then checks its session cache for a match.





If a match is found, and the server is willing to re-establish the connection under the specified session state, it will send a ServerHello with the same Session ID value. At this point, both client and server MUST send change cipher spec messages and proceed directly to finished messages. Once the re-establishment is complete, the client and server MAY begin to exchange application layer data. (See flow chart below.) If a Session ID match is not found, the server generates a new session ID and the TLS client and server perform a full handshake.

```
   Client                                               Server

   ClientHello                 -------->
                                                   ServerHello
                                              [ChangeCipherSpec]
                               <--------             Finished
   [ChangeCipherSpec]
   Finished                    -------->
   Application Data            <-------->     Application Data
```

Fig. 2. Message flow for an abbreviated handshake

The contents and significance of each message will be presented in detail in the following sections.

### 7.4. Handshake Protocol

The TLS Handshake Protocol is one of the defined higher-level clients of the TLS Record Protocol. This protocol is used to negotiate the secure attributes of a session. Handshake messages are supplied to the TLS Record Layer, where they are encapsulated within one or more TLSPlaintext structures, which are processed and transmitted as specified by the current active session state.

```
   enum {
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 986
Keysight, EX-1045
Page 986

```
          hello_request(0), client_hello(1), server_hello(2),
          certificate(11), server_key_exchange (12),
          certificate_request(13), server_hello_done(14),
          certificate_verify(15), client_key_exchange(16),
          finished(20), (255)
      } HandshakeType;

      struct {
          HandshakeType msg_type;    /* handshake type */
          uint24 length;             /* bytes in message */
          select (HandshakeType) {
              case hello_request:       HelloRequest;
              case client_hello:        ClientHello;
```





```
              case server_hello:        ServerHello;
              case certificate:         Certificate;
              case server_key_exchange: ServerKeyExchange;
              case certificate_request: CertificateRequest;
              case server_hello_done:   ServerHelloDone;
              case certificate_verify:  CertificateVerify;
              case client_key_exchange: ClientKeyExchange;
              case finished:            Finished;
          } body;
      } Handshake;
```

The handshake protocol messages are presented below in the order they MUST be sent; sending handshake messages in an unexpected order results in a fatal error. Unneeded handshake messages can be omitted, however. Note one exception to the ordering: the Certificate message is used twice in the handshake (from server to client, then from client to server), but is described only in its first position. The one message that is not bound by these ordering rules is the Hello Request message, which can be sent at any time, but which should be ignored by the client if it arrives in the middle of a handshake.

New Handshake message type values MUST be defined via RFC 2434 Standards Action. See Section 11 for IANA Considerations for these values.

### 7.4.1. Hello Messages

The hello phase messages are used to exchange security enhancement capabilities between the client and server. When a new session begins, the Record Layer's connection state encryption, hash, and compression algorithms are initialized to null. The current connection state is used for renegotiation messages.

#### 7.4.1.1. Hello request

When this message will be sent:

The hello request message MAY be sent by the server at any time.

Meaning of this message:

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

Hello request is a simple notification that the client should begin the negotiation process anew by sending a client hello message when convenient. This message will be ignored by the client if the client is currently negotiating a session. This message may be ignored by the client if it does not wish to renegotiate a session, or the client may, if it wishes, respond





with a no_renegotiation alert. Since handshake messages are intended to have transmission precedence over application data, it is expected that the negotiation will begin before no more than a few records are received from the client. If the server sends a hello request but does not receive a client hello in response, it may close the connection with a fatal alert.

After sending a hello request, servers SHOULD not repeat the request until the subsequent handshake negotiation is complete.

Structure of this message:

```
struct { } HelloRequest;
```

Note: This message MUST NOT be included in the message hashes that are maintained throughout the handshake and used in the finished messages and the certificate verify message.

#### 7.4.1.2. Client Hello

When this message will be sent:

When a client first connects to a server it is required to send the client hello as its first message. The client can also send a client hello in response to a hello request or on its own initiative in order to renegotiate the security parameters in an existing connection.

Structure of this message:

The client hello message includes a random structure, which is used later in the protocol.

```
struct {
   uint32 gmt_unix_time;
   opaque random_bytes[28];
} Random;
```

gmt_unix_time The current time and date in standard UNIX 32-bit format (seconds since the midnight starting Jan 1, 1970, GMT, ignoring leap seconds) according to the sender's internal clock. Clocks are not required to be set correctly by the basic TLS Protocol; higher-level or application protocols may define additional requirements.

random_bytes
28 bytes generated by a secure random number generator.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

The client hello message includes a variable-length session identifier. If not empty, the value identifies a session between the same client and server whose security parameters the client wishes to reuse. The session identifier MAY be from an earlier connection, from this connection, or from another currently active connection. The second option is useful if the client only wishes to update the random structures and derived values of a connection, and the third option makes it possible to establish several independent secure connections without repeating the full handshake protocol. These independent connections may occur sequentially or simultaneously; a SessionID becomes valid when the handshake negotiating it completes with the exchange of Finished messages and persists until it is removed due to aging or because a fatal error was encountered on a connection associated with the session. The actual contents of the SessionID are defined by the server.

```
opaque SessionID<0..32>;
```

Warning: Because the SessionID is transmitted without encryption or immediate MAC protection, servers MUST not place confidential information in session identifiers or let the contents of fake session identifiers cause any breach of security. (Note that the content of the handshake as a whole, including the SessionID, is protected by the Finished messages exchanged at the end of the handshake.)

The CipherSuite list, passed from the client to the server in the client hello message, contains the combinations of cryptographic algorithms supported by the client in order of the client's preference (favorite choice first). Each CipherSuite defines a key exchange algorithm, a bulk encryption algorithm (including secret key length), and a MAC algorithm. The server will select a cipher suite or, if no acceptable choices are presented, return a handshake failure alert and close the connection.

```
uint8 CipherSuite[2];    /* Cryptographic suite selector */
```

The client hello includes a list of compression algorithms supported by the client, ordered according to the client's preference.



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

```
      enum { null(0), (255) } CompressionMethod;

      struct {
          ProtocolVersion client_version;
          Random random;
          SessionID session_id;
          CipherSuite cipher_suites<2..2^16-1>;
          CompressionMethod compression_methods<1..2^8-1>;
      } ClientHello;
```

client_version
   The version of the TLS protocol by which the client wishes to communicate during this session. This SHOULD be the latest (highest valued) version supported by the client. For this version of the specification, the version will be 3.2. (See Appendix E for details about backward compatibility.)

random
   A client-generated random structure.

session_id
   The ID of a session the client wishes to use for this connection. This field should be empty if no session_id is available or if the client wishes to generate new security parameters.

cipher_suites
   This is a list of the cryptographic options supported by the client, with the client's first preference first. If the session_id field is not empty (implying a session resumption request) this vector MUST include at least the cipher_suite from that session. Values are defined in Appendix A.5.

compression_methods
   This is a list of the compression methods supported by the client, sorted by client preference. If the session_id field is not empty (implying a session resumption request) it MUST include the compression_method from that session. This vector MUST contain, and all implementations MUST support, CompressionMethod.null. Thus, a client and server will always be able to agree on a compression method.

After sending the client hello message, the client waits for a server hello message. Any other handshake message returned by the server except for a hello request is treated as a fatal error.

Forward compatibility note: In the interests of forward compatibility, it is permitted that a client hello message include extra data after the compression methods. This data MUST be included





in the handshake hashes, but must otherwise be ignored. This is the only handshake message for which this is legal; for all other messages, the amount of data in the message MUST match the description of the message precisely.

   Note: For the intended use of trailing data in the ClientHello, see RFC 3546 [TLSEXT].

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

7.4.1.3. Server Hello

The server will send this message in response to a client hello message when it was able to find an acceptable set of algorithms. If it cannot find such a match, it will respond with a handshake failure alert.

Structure of this message:

```
struct {
    ProtocolVersion server_version;
    Random random;
    SessionID session_id;
    CipherSuite cipher_suite;
    CompressionMethod compression_method;
} ServerHello;
```

server_version
This field will contain the lower of that suggested by the client in the client hello and the highest supported by the server. For this version of the specification, the version is 3.2. (See Appendix E for details about backward compatibility.)

random
This structure is generated by the server and MUST be independently generated from the ClientHello.random.





session_id
This is the identity of the session corresponding to this connection. If the ClientHello.session_id was non-empty, the server will look in its session cache for a match. If a match is found and the server is willing to establish the new connection using the specified session state, the server will respond with the same value as was supplied by the client. This indicates a resumed session and dictates that the parties must proceed directly to the finished messages. Otherwise this field will contain a different value identifying the new session. The server may return an empty session_id to indicate that the session will not be cached and therefore cannot be resumed. If a session is resumed, it must be resumed using the same cipher suite it was originally negotiated with.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

cipher_suite
The single cipher suite selected by the server from the list in ClientHello.cipher_suites. For resumed sessions, this field is the value from the state of the session being resumed.

compression_method The single compression algorithm selected by the server from the list in ClientHello.compression_methods. For resumed sessions this field is the value from the resumed session state.

7.4.2. Server Certificate

When this message will be sent:

The server MUST send a certificate whenever the agreed-upon key exchange method is not an anonymous one. This message will always immediately follow the server hello message.

Meaning of this message:

The certificate type MUST be appropriate for the selected cipher suite's key exchange algorithm, and is generally an X.509v3 certificate. It MUST contain a key that matches the key exchange method, as follows. Unless otherwise specified, the signing algorithm for the certificate MUST be the same as the algorithm for the certificate key. Unless otherwise specified, the public key MAY be of any length.





| Key Exchange Algorithm | Certificate Key Type |
|---|---|
| RSA | RSA public key; the certificate MUST allow the key to be used for encryption. |
| DHE_DSS | DSS public key. |
| DHE_RSA | RSA public key that can be used for signing. |
| DH_DSS | Diffie-Hellman key. The algorithm used to sign the certificate MUST be DSS. |
| DH_RSA | Diffie-Hellman key. The algorithm used to sign the certificate MUST be RSA. |

All certificate profiles and key and cryptographic formats are defined by the IETF PKIX working group [PKIX]. When a key usage extension is present, the digitalSignature bit MUST be set for the key to be eligible for signing, as described above, and the keyEncipherment bit MUST be present to allow encryption, as described

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

above. The keyAgreement bit must be set on Diffie-Hellman certificates.

As CipherSuites that specify new key exchange methods are specified for the TLS Protocol, they will imply certificate format and the required encoded keying information.

Structure of this message:

```
opaque ASN.1Cert<1..2^24-1>;

struct {
    ASN.1Cert certificate_list<0..2^24-1>;
} Certificate;
```

certificate_list
This is a sequence (chain) of X.509v3 certificates. The sender's certificate must come first in the list. Each following certificate must directly certify the one preceding it. Because certificate validation requires that root keys be distributed independently, the self-signed certificate that specifies the root certificate authority may optionally be omitted from the chain, under the assumption that the remote end must already possess it in order to validate it in any case.

The same message type and structure will be used for the client's response to a certificate request message. Note that a client MAY





send no certificates if it does not have an appropriate certificate to send in response to the server's authentication request.

Note: PKCS #7 [PKCS7] is not used as the format for the certificate vector because PKCS #6 [PKCS6] extended certificates are not used. Also, PKCS #7 defines a SET rather than a SEQUENCE, making the task of parsing the list more difficult.

### 7.4.3. Server Key Exchange Message

When this message will be sent:

This message will be sent immediately after the server certificate message (or the server hello message, if this is an anonymous negotiation).

The server key exchange message is sent by the server only when the server certificate message (if sent) does not contain enough data to allow the client to exchange a premaster secret. This is true for the following key exchange methods:

DHE_DSS
DHE_RSA
DH_anon

It is not legal to send the server key exchange message for the following key exchange methods:

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

```
            RSA
            DH_DSS
            DH_RSA
```

Meaning of this message:

This message conveys cryptographic information to allow the client to communicate the premaster secret: either an RSA public key with which to encrypt the premaster secret, or a Diffie-Hellman public key with which the client can complete a key exchange (with the result being the premaster secret).

As additional CipherSuites are defined for TLS that include new key exchange algorithms, the server key exchange message will be sent if and only if the certificate type associated with the key exchange algorithm does not provide enough information for the client to exchange a premaster secret.





Structure of this message:

```
enum { rsa, diffie_hellman } KeyExchangeAlgorithm;

struct {
    opaque rsa_modulus<1..2^16-1>;
    opaque rsa_exponent<1..2^16-1>;
} ServerRSAParams;
```

rsa_modulus
The modulus of the server's temporary RSA key.

rsa_exponent
The public exponent of the server's temporary RSA key.

```
struct {
    opaque dh_p<1..2^16-1>;
    opaque dh_g<1..2^16-1>;
    opaque dh_Ys<1..2^16-1>;
} ServerDHParams;     /* Ephemeral DH parameters */
```

dh_p
The prime modulus used for the Diffie-Hellman operation.

dh_g
The generator used for the Diffie-Hellman operation.

dh_Ys
The server's Diffie-Hellman public value (g^X mod p).

```
struct {
    select (KeyExchangeAlgorithm) {
        case diffie_hellman:
            ServerDHParams params;
            Signature signed_params;
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 994
Keysight, EX-1045
Page 994

```
            case rsa:
                ServerRSAParams params;
                Signature signed_params;
        };
    } ServerKeyExchange;
```



```
    struct {
        select (KeyExchangeAlgorithm) {
            case diffie_hellman:
                ServerDHParams params;
            case rsa:
                ServerRSAParams params;
        };
     } ServerParams;
```

params
The server's key exchange parameters.

signed_params
For non-anonymous key exchanges, a hash of the corresponding params value, with the signature appropriate to that hash applied.

md5_hash
MD5(ClientHello.random + ServerHello.random + ServerParams);

sha_hash
SHA(ClientHello.random + ServerHello.random + ServerParams);

```
    enum { anonymous, rsa, dsa } SignatureAlgorithm;


    struct {
        select (SignatureAlgorithm) {
            case anonymous: struct { };
            case rsa:
                digitally-signed struct {
                    opaque md5_hash[16];
                    opaque sha_hash[20];
                };
            case dsa:
                digitally-signed struct {
                    opaque sha_hash[20];
                };
            };
        };
    } Signature;
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

### 7.4.4. Certificate request

When this message will be sent:

A non-anonymous server can optionally request a certificate from the client, if it is appropriate for the selected cipher suite.



This message, if sent, will immediately follow the Server Key Exchange message (if it is sent; otherwise, the Server Certificate message).

Structure of this message:

```
enum {
    rsa_sign(1), dss_sign(2), rsa_fixed_dh(3), dss_fixed_dh(4),
 rsa_ephemeral_dh_RESERVED(5), dss_ephemeral_dh_RESERVED(6),
 fortezza_dms_RESERVED(20),
    (255)

} ClientCertificateType;

opaque DistinguishedName<1..2^16-1>;

struct {
    ClientCertificateType certificate_types<1..2^8-1>;
    DistinguishedName certificate_authorities<0..2^16-1>;
} CertificateRequest;
```

certificate_types
This field is a list of the types of certificates requested, sorted in order of the server's preference.

certificate_authorities
A list of the distinguished names of acceptable certificate authorities. These distinguished names may specify a desired distinguished name for a root CA or for a subordinate CA; thus, this message can be used to describe both known roots and a desired authorization space. If the certificate_authorities list is empty then the client MAY send any certificate of the appropriate ClientCertificateType, unless there is some external arrangement to the contrary.

ClientCertificateType values are divided into three groups:

1. Values from 0 (zero) through 63 decimal (0x3F) inclusive are reserved for IETF Standards Track protocols.

2. Values from 64 decimal (0x40) through 223 decimal (0xDF) inclusive are reserved for assignment for non-Standards Track methods.

3. Values from 224 decimal (0xE0) through 255 decimal (0xFF) inclusive are reserved for private use.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

Additional information describing the role of IANA in the allocation of ClientCertificateType code points is described in Section 11.

Note: Values listed as RESERVED may not be used. They were used in SSLv3.

Note: DistinguishedName is derived from [X501]. DistinguishedNames are represented in DER-encoded format.

Note: It is a fatal handshake_failure alert for an anonymous server to request client authentication.

### 7.4.5. Server Hello Done

When this message will be sent:

The server hello done message is sent by the server to indicate the end of the server hello and associated messages. After sending this message, the server will wait for a client response.

Meaning of this message:

This message means that the server is done sending messages to support the key exchange, and the client can proceed with its phase of the key exchange.

Upon receipt of the server hello done message, the client SHOULD verify that the server provided a valid certificate, if required and check that the server hello parameters are acceptable.

Structure of this message:

```
struct { } ServerHelloDone;
```

### 7.4.6. Client certificate

When this message will be sent:

This is the first message the client can send after receiving a server hello done message. This message is only sent if the server requests a certificate. If no suitable certificate is available, the client SHOULD send a certificate message containing no certificates. That is, the certificate_list structure has a length of zero. If client authentication is required by the server for the handshake to continue, it may respond with a fatal handshake failure alert. Client certificates are sent using the Certificate structure defined in Section 7.4.2.



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

Note: When using a static Diffie-Hellman based key exchange method (DH_DSS or DH_RSA), if client authentication is requested, the Diffie-Hellman group and generator encoded in the client's certificate MUST match the server specified Diffie-Hellman parameters if the client's parameters are to be used for the key exchange.

#### 7.4.7. Client Key Exchange Message

When this message will be sent:

This message is always sent by the client. It MUST immediately follow the client certificate message, if it is sent. Otherwise it MUST be the first message sent by the client after it receives the server hello done message.

Meaning of this message:

With this message, the premaster secret is set, either though direct transmission of the RSA-encrypted secret or by the transmission of Diffie-Hellman parameters that will allow each side to agree upon the same premaster secret. When the key exchange method is DH_RSA or DH_DSS, client certification has been requested, and the client was able to respond with a certificate that contained a Diffie-Hellman public key whose parameters (group and generator) matched those specified by the server in its certificate, this message MUST not contain any data.

Structure of this message:

The choice of messages depends on which key exchange method has been selected. See Section 7.4.3 for the KeyExchangeAlgorithm definition.

```
struct {
    select (KeyExchangeAlgorithm) {
        case rsa: EncryptedPreMasterSecret;
        case diffie_hellman: ClientDiffieHellmanPublic;
    } exchange_keys;
} ClientKeyExchange;
```

##### 7.4.7.1. RSA Encrypted Premaster Secret Message

Meaning of this message:

If RSA is being used for key agreement and authentication, the client generates a 48-byte premaster secret, encrypts it using the public key from the server's certificate or the temporary RSA key



provided in a server key exchange message, and sends the result in an encrypted premaster secret message. This structure is a variant of the client key exchange message and is not a message in itself.

Structure of this message:

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 998
Keysight, EX-1045
Page 998

```
      struct {
          ProtocolVersion client_version;
          opaque random[46];
      } PreMasterSecret;

      client_version The latest (newest) version supported by the
         client.  This is used to detect version roll-back attacks.
         Upon receiving the premaster secret, the server SHOULD check
         that this value matches the value transmitted by the client in
         the client hello message.

      random
          46 securely-generated random bytes.

      struct {
          public-key-encrypted PreMasterSecret pre_master_secret;
      } EncryptedPreMasterSecret;

      pre_master_secret
          This random value is generated by the client and is used to
          generate the master secret, as specified in Section 8.1.

   Note: An attack discovered by Daniel Bleichenbacher [BLEI] can be
         used to attack a TLS server that is using PKCS#1 v 1.5 encoded
         RSA.  The attack takes advantage of the fact that, by failing
         in different ways, a TLS server can be coerced into revealing
         whether a particular message, when decrypted, is properly
         PKCS#1 v1.5 formatted or not.

         The best way to avoid vulnerability to this attack is to treat
         incorrectly formatted messages in a manner indistinguishable
         from correctly formatted RSA blocks.  Thus, when a server
         receives an incorrectly formatted RSA block, it should generate
         a random 48-byte value and proceed using it as the premaster
         secret.  Thus, the server will act identically whether the
         received RSA block is correctly encoded or not.

         [PKCS1B] defines a newer version of PKCS#1 encoding that is
         more secure against the Bleichenbacher attack.  However, for
         maximal compatibility with TLS 1.0, TLS 1.1 retains the
         original encoding.  No variants of the Bleichenbacher attack
```





```
         are known to exist provided that the above recommendations are
         followed.

   Implementation Note: Public-key-encrypted data is represented as an
                        opaque vector <0..2^16-1> (see Section 4.7).
                        Thus, the RSA-encrypted PreMasterSecret in a
                        ClientKeyExchange is preceded by two length
                        bytes.  These bytes are redundant in the case of
                        RSA because the EncryptedPreMasterSecret is the
                        only data in the ClientKeyExchange and its
                        length can therefore be unambiguously
                        determined.  The SSLv3 specification was not
                        clear about the encoding of public-key-encrypted
                        data, and therefore many SSLv3 implementations
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

do not include the length bytes, encoding the RSA encrypted data directly in the ClientKeyExchange message.

This specification requires correct encoding of the EncryptedPreMasterSecret complete with length bytes. The resulting PDU is incompatible with many SSLv3 implementations. Implementors upgrading from SSLv3 must modify their implementations to generate and accept the correct encoding. Implementors who wish to be compatible with both SSLv3 and TLS should make their implementation's behavior dependent on the protocol version.

Implementation Note: It is now known that remote timing-based attacks on SSL are possible, at least when the client and server are on the same LAN. Accordingly, implementations that use static RSA keys SHOULD use RSA blinding or some other anti-timing technique, as described in [TIMING].

Note: The version number in the PreMasterSecret MUST be the version offered by the client in the ClientHello, not the version negotiated for the connection. This feature is designed to prevent rollback attacks. Unfortunately, many implementations use the negotiated version instead, and therefore checking the version number may lead to failure to interoperate with such incorrect client implementations. Client implementations, MUST and Server implementations MAY, check the version number. In practice, since the TLS handshake MACs prevent downgrade and no good attacks are known on those MACs, ambiguity is not considered a serious security risk. Note that if servers choose to check the version number, they should randomize the





PreMasterSecret in case of error, rather than generate an alert, in order to avoid variants on the Bleichenbacher attack. [KPR03]

#### 7.4.7.2. Client Diffie-Hellman Public Value

Meaning of this message:

This structure conveys the client's Diffie-Hellman public value (Yc) if it was not already included in the client's certificate. The encoding used for Yc is determined by the enumerated PublicValueEncoding. This structure is a variant of the client key exchange message and not a message in itself.

Structure of this message:

```
enum { implicit, explicit } PublicValueEncoding;

implicit
    If the client certificate already contains a suitable Diffie-
    Hellman key, then Yc is implicit and does not need to be sent
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1000
Keysight, EX-1045
Page 1000

```
            again.  In this case, the client key exchange message will be
            sent, but it MUST be empty.

        explicit
            Yc needs to be sent.

        struct {
            select (PublicValueEncoding) {
                case implicit: struct { };
                case explicit: opaque dh_Yc<1..2^16-1>;
            } dh_public;
        } ClientDiffieHellmanPublic;

        dh_Yc
            The client's Diffie-Hellman public value (Yc).
```

### 7.4.8. Certificate verify

When this message will be sent:

This message is used to provide explicit verification of a client certificate. This message is only sent following a client certificate that has signing capability (i.e., all certificates except those containing fixed Diffie-Hellman parameters). When sent, it MUST immediately follow the client key exchange message.





Structure of this message:

```
        struct {
             Signature signature;
        } CertificateVerify;
```

The Signature type is defined in 7.4.3.

```
        CertificateVerify.signature.md5_hash
            MD5(handshake_messages);

        CertificateVerify.signature.sha_hash
            SHA(handshake_messages);
```

Here handshake_messages refers to all handshake messages sent or received starting at client hello up to but not including this message, including the type and length fields of the handshake messages. This is the concatenation of all the Handshake structures, as defined in 7.4, exchanged thus far.

### 7.4.9. Finished

When this message will be sent:

A finished message is always sent immediately after a change cipher spec message to verify that the key exchange and authentication processes were successful. It is essential that a change cipher spec message be received between the other handshake

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

messages and the Finished message.

Meaning of this message:

The finished message is the first protected with the just-negotiated algorithms, keys, and secrets. Recipients of finished messages MUST verify that the contents are correct. Once a side has sent its Finished message and received and validated the Finished message from its peer, it may begin to send and receive application data over the connection.

```
struct {
    opaque verify_data[12];
} Finished;

verify_data
    PRF(master_secret, finished_label, MD5(handshake_messages) +
    SHA-1(handshake_messages)) [0..11];
```





finished_label
: For Finished messages sent by the client, the string "client finished". For Finished messages sent by the server, the string "server finished".

handshake_messages
: All of the data from all messages in this handshake (not including any HelloRequest messages) up to but not including this message. This is only data visible at the handshake layer and does not include record layer headers. This is the concatenation of all the Handshake structures, as defined in 7.4, exchanged thus far.

It is a fatal error if a finished message is not preceded by a change cipher spec message at the appropriate point in the handshake.

The value handshake_messages includes all handshake messages starting at client hello up to, but not including, this finished message. This may be different from handshake_messages in Section 7.4.8 because it would include the certificate verify message (if sent). Also, the handshake_messages for the finished message sent by the client will be different from that for the finished message sent by the server, because the one that is sent second will include the prior one.

Note: Change cipher spec messages, alerts, and any other record types are not handshake messages and are not included in the hash computations. Also, Hello Request messages are omitted from handshake hashes.

## 8. Cryptographic Computations

In order to begin connection protection, the TLS Record Protocol requires specification of a suite of algorithms, a master secret, and the client and server random values. The authentication, encryption,

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

and MAC algorithms are determined by the cipher_suite selected by the server and revealed in the server hello message. The compression algorithm is negotiated in the hello messages, and the random values are exchanged in the hello messages. All that remains is to calculate the master secret.

### 8.1. Computing the Master Secret

For all key exchange methods, the same algorithm is used to convert the pre_master_secret into the master_secret. The pre_master_secret should be deleted from memory once the master_secret has been computed.





```
master_secret = PRF(pre_master_secret, "master secret",
                    ClientHello.random + ServerHello.random)
[0..47];
```

The master secret is always exactly 48 bytes in length. The length of the premaster secret will vary depending on key exchange method.

#### 8.1.1. RSA

When RSA is used for server authentication and key exchange, a 48-byte pre_master_secret is generated by the client, encrypted under the server's public key, and sent to the server. The server uses its private key to decrypt the pre_master_secret. Both parties then convert the pre_master_secret into the master_secret, as specified above.

RSA digital signatures are performed using PKCS #1 [PKCS1] block type 1. RSA public key encryption is performed using PKCS #1 block type 2.

#### 8.1.2. Diffie-Hellman

A conventional Diffie-Hellman computation is performed. The negotiated key (Z) is used as the pre_master_secret, and is converted into the master_secret, as specified above. Leading bytes of Z that contain all zero bits are stripped before it is used as the pre_master_secret.

Note: Diffie-Hellman parameters are specified by the server and may be either ephemeral or contained within the server's certificate.

## 9. Mandatory Cipher Suites

In the absence of an application profile standard specifying otherwise, a TLS compliant application MUST implement the cipher suite TLS_RSA_WITH_3DES_EDE_CBC_SHA.

## 10. Application Data Protocol

Application data messages are carried by the Record Layer and are fragmented, compressed, and encrypted based on the current connection state. The messages are treated as transparent data to the record

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

layer.

## 11. Security Considerations

Security issues are discussed throughout this memo, especially in Appendices D, E, and F.





## 12. IANA Considerations

This document describes a number of new registries that have been created by IANA. We recommended that they be placed as individual registries items under a common TLS category.

Section 7.4.3 describes a TLS ClientCertificateType Registry to be maintained by the IANA, defining a number of such code point identifiers. ClientCertificateType identifiers with values in the range 0-63 (decimal) inclusive are assigned via RFC 2434 Standards Action. Values from the range 64-223 (decimal) inclusive are assigned via [RFC2434] Specification Required. Identifier values from 224-255 (decimal) inclusive are reserved for RFC 2434 Private Use. The registry will initially be populated with the values in this document, Section 7.4.4.

Section A.5 describes a TLS Cipher Suite Registry to be maintained by the IANA, and it defines a number of such cipher suite identifiers. Cipher suite values with the first byte in the range 0-191 (decimal) inclusive are assigned via RFC 2434 Standards Action. Values with the first byte in the range 192-254 (decimal) are assigned via RFC 2434 Specification Required. Values with the first byte 255 (decimal) are reserved for RFC 2434 Private Use. The registry will initially be populated with the values from Section A.5 of this document, [TLSAES], and from Section 3 of [TLSKRB].

Section 6 requires that all ContentType values be defined by RFC 2434 Standards Action. IANA has created a TLS ContentType registry, initially populated with values from Section 6.2.1 of this document. Future values MUST be allocated via Standards Action as described in [RFC2434].

Section 7.2.2 requires that all Alert values be defined by RFC 2434 Standards Action. IANA has created a TLS Alert registry, initially populated with values from Section 7.2 of this document and from Section 4 of [TLSEXT]. Future values MUST be allocated via Standards Action as described in [RFC2434].

Section 7.4 requires that all HandshakeType values be defined by RFC 2434 Standards Action. IANA has created a TLS HandshakeType registry, initially populated with values from Section 7.4 of this document and from Section 2.4 of [TLSEXT]. Future values MUST be allocated via Standards Action as described in [RFC2434].

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

## Appendix A. Protocol Constant Values

This section describes protocol types and constants.

### A.1. Record Layer

```
struct {
    uint8 major, minor;
} ProtocolVersion;

ProtocolVersion version = { 3, 2 };     /* TLS v1.1 */

enum {
    change_cipher_spec(20), alert(21), handshake(22),
    application_data(23), (255)
} ContentType;

struct {
    ContentType type;
    ProtocolVersion version;
    uint16 length;
    opaque fragment[TLSPlaintext.length];
} TLSPlaintext;

struct {
    ContentType type;
    ProtocolVersion version;
    uint16 length;
    opaque fragment[TLSCompressed.length];
} TLSCompressed;

struct {
    ContentType type;
    ProtocolVersion version;
    uint16 length;
    select (CipherSpec.cipher_type) {
        case stream: GenericStreamCipher;
        case block:  GenericBlockCipher;
    } fragment;
} TLSCiphertext;

stream-ciphered struct {
    opaque content[TLSCompressed.length];
    opaque MAC[CipherSpec.hash_size];
} GenericStreamCipher;

block-ciphered struct {
    opaque IV[CipherSpec.block_length];
```



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

```
        opaque content[TLSCompressed.length];
        opaque MAC[CipherSpec.hash_size];
        uint8 padding[GenericBlockCipher.padding_length];
        uint8 padding_length;
    } GenericBlockCipher;
```

### A.2. Change Cipher Specs Message

```
    struct {
        enum { change_cipher_spec(1), (255) } type;
    } ChangeCipherSpec;
```

### A.3. Alert Messages

```
    enum { warning(1), fatal(2), (255) } AlertLevel;

        enum {
            close_notify(0),
            unexpected_message(10),
            bad_record_mac(20),
            decryption_failed(21),
            record_overflow(22),
            decompression_failure(30),
            handshake_failure(40),
            no_certificate_RESERVED (41),
            bad_certificate(42),
            unsupported_certificate(43),
            certificate_revoked(44),
            certificate_expired(45),
            certificate_unknown(46),
            illegal_parameter(47),
            unknown_ca(48),
            access_denied(49),
            decode_error(50),
            decrypt_error(51),
            export_restriction_RESERVED(60),
            protocol_version(70),
            insufficient_security(71),
            internal_error(80),
            user_canceled(90),
            no_renegotiation(100),
            (255)
        } AlertDescription;

    struct {
        AlertLevel level;
        AlertDescription description;
    } Alert;
```





### A.4. Handshake Protocol

```
    enum {
        hello_request(0), client_hello(1), server_hello(2),
        certificate(11), server_key_exchange (12),
        certificate_request(13), server_hello_done(14),
        certificate_verify(15), client_key_exchange(16),
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

```
        finished(20), (255)
    } HandshakeType;

    struct {
        HandshakeType msg_type;
        uint24 length;
        select (HandshakeType) {
            case hello_request:       HelloRequest;
            case client_hello:        ClientHello;
            case server_hello:        ServerHello;
            case certificate:         Certificate;
            case server_key_exchange: ServerKeyExchange;
            case certificate_request: CertificateRequest;
            case server_hello_done:   ServerHelloDone;
            case certificate_verify:  CertificateVerify;
            case client_key_exchange: ClientKeyExchange;
            case finished:            Finished;
        } body;
    } Handshake;
```

### A.4.1. Hello messages

```
    struct { } HelloRequest;

    struct {
        uint32 gmt_unix_time;
        opaque random_bytes[28];
    } Random;

    opaque SessionID<0..32>;

    uint8 CipherSuite[2];

    enum { null(0), (255) } CompressionMethod;

    struct {
        ProtocolVersion client_version;
        Random random;
        SessionID session_id;
        CipherSuite cipher_suites<2..2^16-1>;
        CompressionMethod compression_methods<1..2^8-1>;
```





```
    } ClientHello;

    struct {
        ProtocolVersion server_version;
        Random random;
        SessionID session_id;
        CipherSuite cipher_suite;
        CompressionMethod compression_method;
    } ServerHello;
```

### A.4.2. Server Authentication and Key Exchange Messages

```
    opaque ASN.1Cert<2^24-1>;
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

```
   struct {
       ASN.1Cert certificate_list<0..2^24-1>;
   } Certificate;

   enum { rsa, diffie_hellman } KeyExchangeAlgorithm;

   struct {
       opaque rsa_modulus<1..2^16-1>;
       opaque rsa_exponent<1..2^16-1>;
   } ServerRSAParams;

   struct {
       opaque dh_p<1..2^16-1>;
       opaque dh_g<1..2^16-1>;
       opaque dh_Ys<1..2^16-1>;
   } ServerDHParams;

   struct {
       select (KeyExchangeAlgorithm) {
           case diffie_hellman:
               ServerDHParams params;
               Signature signed_params;
           case rsa:
               ServerRSAParams params;
               Signature signed_params;
       };
   } ServerKeyExchange;

   enum { anonymous, rsa, dsa } SignatureAlgorithm;

   struct {
       select (KeyExchangeAlgorithm) {
           case diffie_hellman:
               ServerDHParams params;
```





```
           case rsa:
               ServerRSAParams params;
       };
   } ServerParams;

   struct {
       select (SignatureAlgorithm) {
           case anonymous: struct { };
           case rsa:
               digitally-signed struct {
                   opaque md5_hash[16];
                   opaque sha_hash[20];
               };
           case dsa:
               digitally-signed struct {
                   opaque sha_hash[20];
               };
           };
       };
   } Signature;
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1008
Keysight, EX-1045
Page 1008

```
   enum {
       rsa_sign(1), dss_sign(2), rsa_fixed_dh(3), dss_fixed_dh(4),
    rsa_ephemeral_dh_RESERVED(5), dss_ephemeral_dh_RESERVED(6),
    fortezza_dms_RESERVED(20),
    (255)
   } ClientCertificateType;

   opaque DistinguishedName<1..2^16-1>;

   struct {
       ClientCertificateType certificate_types<1..2^8-1>;
       DistinguishedName certificate_authorities<0..2^16-1>;
   } CertificateRequest;

   struct { } ServerHelloDone;
```

A.4.3. Client Authentication and Key Exchange Messages

```
   struct {
       select (KeyExchangeAlgorithm) {
           case rsa: EncryptedPreMasterSecret;
           case diffie_hellman: ClientDiffieHellmanPublic;
       } exchange_keys;
   } ClientKeyExchange;
```





```
   struct {
       ProtocolVersion client_version;
       opaque random[46];
   }
   PreMasterSecret;

   struct {
       public-key-encrypted PreMasterSecret pre_master_secret;
   } EncryptedPreMasterSecret;

   enum { implicit, explicit } PublicValueEncoding;

   struct {
       select (PublicValueEncoding) {
           case implicit: struct {};
           case explicit: opaque DH_Yc<1..2^16-1>;
       } dh_public;
   } ClientDiffieHellmanPublic;

   struct {
       Signature signature;
   } CertificateVerify;
```

A.4.4. Handshake Finalization Message

```
   struct {
       opaque verify_data[12];
   } Finished;
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

## A.5. The CipherSuite

The following values define the CipherSuite codes used in the client hello and server hello messages.

A CipherSuite defines a cipher specification supported in TLS Version 1.1.

TLS_NULL_WITH_NULL_NULL is specified and is the initial state of a TLS connection during the first handshake on that channel, but must not be negotiated, as it provides no more protection than an unsecured connection.

```
 CipherSuite TLS_NULL_WITH_NULL_NULL               = { 0x00,0x00 };
```

The following CipherSuite definitions require that the server provide an RSA certificate that can be used for key exchange. The server may request either an RSA or a DSS signature-capable certificate in the certificate request message.





```
 CipherSuite TLS_RSA_WITH_NULL_MD5                 = { 0x00,0x01 };
 CipherSuite TLS_RSA_WITH_NULL_SHA                 = { 0x00,0x02 };
 CipherSuite TLS_RSA_WITH_RC4_128_MD5              = { 0x00,0x04 };
 CipherSuite TLS_RSA_WITH_RC4_128_SHA              = { 0x00,0x05 };
 CipherSuite TLS_RSA_WITH_IDEA_CBC_SHA             = { 0x00,0x07 };
 CipherSuite TLS_RSA_WITH_DES_CBC_SHA              = { 0x00,0x09 };
 CipherSuite TLS_RSA_WITH_3DES_EDE_CBC_SHA         = { 0x00,0x0A };
```

The following CipherSuite definitions are used for server-authenticated (and optionally client-authenticated) Diffie-Hellman. DH denotes cipher suites in which the server's certificate contains the Diffie-Hellman parameters signed by the certificate authority (CA). DHE denotes ephemeral Diffie-Hellman, where the Diffie-Hellman parameters are signed by a DSS or RSA certificate that has been signed by the CA. The signing algorithm used is specified after the DH or DHE parameter. The server can request an RSA or DSS signature-capable certificate from the client for client authentication or it may request a Diffie-Hellman certificate. Any Diffie-Hellman certificate provided by the client must use the parameters (group and generator) described by the server.

```
 CipherSuite TLS_DH_DSS_WITH_DES_CBC_SHA           = { 0x00,0x0C };
 CipherSuite TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA      = { 0x00,0x0D };
 CipherSuite TLS_DH_RSA_WITH_DES_CBC_SHA           = { 0x00,0x0F };
 CipherSuite TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA      = { 0x00,0x10 };
 CipherSuite TLS_DHE_DSS_WITH_DES_CBC_SHA          = { 0x00,0x12 };
 CipherSuite TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA     = { 0x00,0x13 };
 CipherSuite TLS_DHE_RSA_WITH_DES_CBC_SHA          = { 0x00,0x15 };
 CipherSuite TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA     = { 0x00,0x16 };
```

The following cipher suites are used for completely anonymous Diffie-Hellman communications in which neither party is authenticated. Note that this mode is vulnerable to man-in-the-middle attacks and is therefore deprecated.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1010
Keysight, EX-1045
Page 1010

```
 CipherSuite TLS_DH_anon_WITH_RC4_128_MD5          = { 0x00,0x18 };
 CipherSuite TLS_DH_anon_WITH_DES_CBC_SHA          = { 0x00,0x1A };
 CipherSuite TLS_DH_anon_WITH_3DES_EDE_CBC_SHA     = { 0x00,0x1B };
```

When SSLv3 and TLS 1.0 were designed, the United States restricted the export of cryptographic software containing certain strong encryption algorithms. A series of cipher suites were designed to operate at reduced key lengths in order to comply with those regulations. Due to advances in computer performance, these algorithms are now unacceptably weak, and export restrictions have since been loosened. TLS 1.1 implementations MUST NOT negotiate these cipher suites in TLS 1.1 mode. However, for backward compatibility they may be offered in the ClientHello for use with TLS





1.0 or SSLv3-only servers. TLS 1.1 clients MUST check that the server did not choose one of these cipher suites during the handshake. These ciphersuites are listed below for informational purposes and to reserve the numbers.

```
 CipherSuite TLS_RSA_EXPORT_WITH_RC4_40_MD5        = { 0x00,0x03 };
 CipherSuite TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5    = { 0x00,0x06 };
 CipherSuite TLS_RSA_EXPORT_WITH_DES40_CBC_SHA     = { 0x00,0x08 };
 CipherSuite TLS_DH_DSS_EXPORT_WITH_DES40_CBC_SHA  = { 0x00,0x0B };
 CipherSuite TLS_DH_RSA_EXPORT_WITH_DES40_CBC_SHA  = { 0x00,0x0E };
 CipherSuite TLS_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA = { 0x00,0x11 };
 CipherSuite TLS_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA = { 0x00,0x14 };
 CipherSuite TLS_DH_anon_EXPORT_WITH_RC4_40_MD5    = { 0x00,0x17 };
 CipherSuite TLS_DH_anon_EXPORT_WITH_DES40_CBC_SHA = { 0x00,0x19 };
```

The following cipher suites were defined in [TLSKRB] and are included here for completeness. See [TLSKRB] for details:

```
 CipherSuite    TLS_KRB5_WITH_DES_CBC_SHA          = { 0x00,0x1E }:
 CipherSuite    TLS_KRB5_WITH_3DES_EDE_CBC_SHA     = { 0x00,0x1F };
 CipherSuite    TLS_KRB5_WITH_RC4_128_SHA          = { 0x00,0x20 };
 CipherSuite    TLS_KRB5_WITH_IDEA_CBC_SHA         = { 0x00,0x21 };
 CipherSuite    TLS_KRB5_WITH_DES_CBC_MD5          = { 0x00,0x22 };
 CipherSuite    TLS_KRB5_WITH_3DES_EDE_CBC_MD5     = { 0x00,0x23 };
 CipherSuite    TLS_KRB5_WITH_RC4_128_MD5          = { 0x00,0x24 };
 CipherSuite    TLS_KRB5_WITH_IDEA_CBC_MD5         = { 0x00,0x25 };
```

The following exportable cipher suites were defined in [TLSKRB] and are included here for completeness. TLS 1.1 implementations MUST NOT negotiate these cipher suites.

```
 CipherSuite  TLS_KRB5_EXPORT_WITH_DES_CBC_40_SHA    = { 0x00,0x26};
 CipherSuite  TLS_KRB5_EXPORT_WITH_RC2_CBC_40_SHA    = { 0x00,0x27};
 CipherSuite  TLS_KRB5_EXPORT_WITH_RC4_40_SHA        = { 0x00,0x28};
 CipherSuite  TLS_KRB5_EXPORT_WITH_DES_CBC_40_MD5    = { 0x00,0x29};
 CipherSuite  TLS_KRB5_EXPORT_WITH_RC2_CBC_40_MD5    = { 0x00,0x2A};
 CipherSuite  TLS_KRB5_EXPORT_WITH_RC4_40_MD5        = { 0x00,0x2B};
```

The following cipher suites were defined in [TLSAES] and are included here for completeness. See [TLSAES] for details:

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1011
Keysight, EX-1045
Page 1011

```
          CipherSuite TLS_RSA_WITH_AES_128_CBC_SHA      = { 0x00, 0x2F };
          CipherSuite TLS_DH_DSS_WITH_AES_128_CBC_SHA   = { 0x00, 0x30 };
          CipherSuite TLS_DH_RSA_WITH_AES_128_CBC_SHA   = { 0x00, 0x31 };
          CipherSuite TLS_DHE_DSS_WITH_AES_128_CBC_SHA  = { 0x00, 0x32 };
          CipherSuite TLS_DHE_RSA_WITH_AES_128_CBC_SHA  = { 0x00, 0x33 };
          CipherSuite TLS_DH_anon_WITH_AES_128_CBC_SHA  = { 0x00, 0x34 };
```



```
          CipherSuite TLS_RSA_WITH_AES_256_CBC_SHA      = { 0x00, 0x35 };
          CipherSuite TLS_DH_DSS_WITH_AES_256_CBC_SHA   = { 0x00, 0x36 };
          CipherSuite TLS_DH_RSA_WITH_AES_256_CBC_SHA   = { 0x00, 0x37 };
          CipherSuite TLS_DHE_DSS_WITH_AES_256_CBC_SHA  = { 0x00, 0x38 };
          CipherSuite TLS_DHE_RSA_WITH_AES_256_CBC_SHA  = { 0x00, 0x39 };
          CipherSuite TLS_DH_anon_WITH_AES_256_CBC_SHA  = { 0x00, 0x3A };
```

The cipher suite space is divided into three regions:

1. Cipher suite values with first byte 0x00 (zero) through decimal 191 (0xBF) inclusive are reserved for the IETF Standards Track protocols.

2. Cipher suite values with first byte decimal 192 (0xC0) through decimal 254 (0xFE) inclusive are reserved for assignment for non-Standards Track methods.

3. Cipher suite values with first byte 0xFF are reserved for private use.

Additional information describing the role of IANA in the allocation of cipher suite code points is described in Section 11.

Note: The cipher suite values { 0x00, 0x1C } and { 0x00, 0x1D } are reserved to avoid collision with Fortezza-based cipher suites in SSL 3.

### A.6. The Security Parameters

These security parameters are determined by the TLS Handshake Protocol and provided as parameters to the TLS Record Layer in order to initialize a connection state. SecurityParameters includes:

```
   enum { null(0), (255) } CompressionMethod;

   enum { server, client } ConnectionEnd;

   enum { null, rc4, rc2, des, 3des, des40, aes, idea }
   BulkCipherAlgorithm;

   enum { stream, block } CipherType;

   enum { null, md5, sha } MACAlgorithm;

/* The algorithms specified in CompressionMethod,
BulkCipherAlgorithm, and MACAlgorithm may be added to. */
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

```
            struct {
                ConnectionEnd entity;
                BulkCipherAlgorithm bulk_cipher_algorithm;
                CipherType cipher_type;
                uint8 key_size;
                uint8 key_material_length;
                MACAlgorithm mac_algorithm;
                uint8 hash_size;
                CompressionMethod compression_algorithm;
                opaque master_secret[48];
                opaque client_random[32];
                opaque server_random[32];
            } SecurityParameters;
```

## Appendix B. Glossary

**Advanced Encryption Standard (AES)**
AES is a widely used symmetric encryption algorithm. AES is a block cipher with a 128, 192, or 256 bit keys and a 16 byte block size. [AES] TLS currently only supports the 128 and 256 bit key sizes.

**application protocol**
An application protocol is a protocol that normally layers directly on top of the transport layer (e.g., TCP/IP). Examples include HTTP, TELNET, FTP, and SMTP.

**asymmetric cipher**
See public key cryptography.

**authentication**
Authentication is the ability of one entity to determine the identity of another entity.

**block cipher**
A block cipher is an algorithm that operates on plaintext in groups of bits, called blocks. 64 bits is a common block size.

**bulk cipher**
A symmetric encryption algorithm used to encrypt large quantities of data.

**cipher block chaining (CBC)**
CBC is a mode in which every plaintext block encrypted with a block cipher is first exclusive-ORed with the previous ciphertext block (or, in the case of the first block, with the initialization vector). For decryption, every block is first decrypted, then exclusive-ORed with the previous ciphertext block (or IV).



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

certificate
   As part of the X.509 protocol (a.k.a. ISO Authentication framework), certificates are assigned by a trusted Certificate Authority and provide a strong binding between a party's identity or some other attributes and its public key.

client
   The application entity that initiates a TLS connection to a server. This may or may not imply that the client initiated the underlying transport connection. The primary operational difference between the server and client is that the server is generally authenticated, while the client is only optionally authenticated.

client write key
   The key used to encrypt data written by the client.

client write MAC secret
   The secret data used to authenticate data written by the client.

connection
   A connection is a transport (in the OSI layering model definition) that provides a suitable type of service. For TLS, such connections are peer-to-peer relationships. The connections are transient. Every connection is associated with one session.

Data Encryption Standard
   DES is a very widely used symmetric encryption algorithm. DES is a block cipher with a 56 bit key and an 8 byte block size. Note that in TLS, for key generation purposes, DES is treated as having an 8 byte key length (64 bits), but it still only provides 56 bits of protection. (The low bit of each key byte is presumed to be set to produce odd parity in that key byte.) DES can also be operated in a mode where three independent keys and three encryptions are used for each block of data; this uses 168 bits of key (24 bytes in the TLS key generation method) and provides the equivalent of 112 bits of security. [DES], [3DES]

Digital Signature Standard (DSS)
   A standard for digital signing, including the Digital Signing Algorithm, approved by the National Institute of Standards and Technology, defined in NIST FIPS PUB 186, "Digital Signature Standard," published May 1994 by the U.S. Dept. of Commerce. [DSS]





digital signatures
   Digital signatures utilize public key cryptography and one-way hash functions to produce a signature of the data that can be authenticated, and is difficult to forge or repudiate.

handshake
   An initial negotiation between client and server that establishes

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

the parameters of their transactions.

Initialization Vector (IV)
When a block cipher is used in CBC mode, the initialization vector is exclusive-ORed with the first plaintext block prior to encryption.

IDEA
A 64-bit block cipher designed by Xuejia Lai and James Massey. [IDEA]

Message Authentication Code (MAC)
A Message Authentication Code is a one-way hash computed from a message and some secret data. It is difficult to forge without knowing the secret data. Its purpose is to detect if the message has been altered.

master secret
Secure secret data used for generating encryption keys, MAC secrets, and IVs.

MD5
MD5 is a secure hashing function that converts an arbitrarily long data stream into a digest of fixed size (16 bytes). [MD5]

public key cryptography
A class of cryptographic techniques employing two-key ciphers. Messages encrypted with the public key can only be decrypted with the associated private key. Conversely, messages signed with the private key can be verified with the public key.

one-way hash function
A one-way transformation that converts an arbitrary amount of data into a fixed-length hash. It is computationally hard to reverse the transformation or to find collisions. MD5 and SHA are examples of one-way hash functions.

RC2
A block cipher developed by Ron Rivest at RSA Data Security, Inc. [RSADSI] described in [RC2].





RC4
A stream cipher invented by Ron Rivest. A compatible cipher is described in [SCH].

RSA
A very widely used public-key algorithm that can be used for either encryption or digital signing. [RSA]

server
The server is the application entity that responds to requests for connections from clients. See also under client.

session
A TLS session is an association between a client and a server.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

Sessions are created by the handshake protocol. Sessions define a set of cryptographic security parameters that can be shared among multiple connections. Sessions are used to avoid the expensive negotiation of new security parameters for each connection.

session identifier
A session identifier is a value generated by a server that identifies a particular session.

server write key
The key used to encrypt data written by the server.

server write MAC secret
The secret data used to authenticate data written by the server.

SHA
The Secure Hash Algorithm is defined in FIPS PUB 180-2. It produces a 20-byte output. Note that all references to SHA actually use the modified SHA-1 algorithm. [SHA]

SSL
Netscape's Secure Socket Layer protocol [SSL3]. TLS is based on SSL Version 3.0

stream cipher
An encryption algorithm that converts a key into a cryptographically strong keystream, which is then exclusive-ORed with the plaintext.

symmetric cipher
See bulk cipher.





Transport Layer Security (TLS)
This protocol; also, the Transport Layer Security working group of the Internet Engineering Task Force (IETF). See "Comments" at the end of this document.

## Appendix C. CipherSuite Definitions

| CipherSuite | Key Exchange | Cipher | Hash |
|---|---|---|---|
| TLS_NULL_WITH_NULL_NULL | NULL | NULL | NULL |
| TLS_RSA_WITH_NULL_MD5 | RSA | NULL | MD5 |
| TLS_RSA_WITH_NULL_SHA | RSA | NULL | SHA |
| TLS_RSA_WITH_RC4_128_MD5 | RSA | RC4_128 | MD5 |
| TLS_RSA_WITH_RC4_128_SHA | RSA | RC4_128 | SHA |
| TLS_RSA_WITH_IDEA_CBC_SHA | RSA | IDEA_CBC | SHA |
| TLS_RSA_WITH_DES_CBC_SHA | RSA | DES_CBC | SHA |
| TLS_RSA_WITH_3DES_EDE_CBC_SHA | RSA | 3DES_EDE_CBC | SHA |
| TLS_DH_DSS_WITH_DES_CBC_SHA | DH_DSS | DES_CBC | SHA |
| TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA | DH_DSS | 3DES_EDE_CBC | SHA |
| TLS_DH_RSA_WITH_DES_CBC_SHA | DH_RSA | DES_CBC | SHA |
| TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA | DH_RSA | 3DES_EDE_CBC | SHA |

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

```
TLS_DHE_DSS_WITH_DES_CBC_SHA          DHE_DSS        DES_CBC      SHA
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA     DHE_DSS        3DES_EDE_CBC SHA
TLS_DHE_RSA_WITH_DES_CBC_SHA          DHE_RSA        DES_CBC      SHA
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA     DHE_RSA        3DES_EDE_CBC SHA
TLS_DH_anon_WITH_RC4_128_MD5          DH_anon        RC4_128      MD5
TLS_DH_anon_WITH_DES_CBC_SHA          DH_anon        DES_CBC      SHA
TLS_DH_anon_WITH_3DES_EDE_CBC_SHA     DH_anon        3DES_EDE_CBC SHA
```

| Key Exchange Algorithm | Description | Key size limit |
|---|---|---|
| DHE_DSS | Ephemeral DH with DSS signatures | None |
| DHE_RSA | Ephemeral DH with RSA signatures | None |
| DH_anon | Anonymous DH, no signatures | None |
| DH_DSS | DH with DSS-based certificates | None |
| DH_RSA | DH with RSA-based certificates | None |
| | | RSA = none |
| NULL | No key exchange | N/A |
| RSA | RSA key exchange | None |





| Cipher | Type | Key Material | Expanded Key Material | IV Size | Block Size |
|---|---|---|---|---|---|
| NULL | Stream | 0 | 0 | 0 | N/A |
| IDEA_CBC | Block | 16 | 16 | 8 | 8 |
| RC2_CBC_40 | Block | 5 | 16 | 8 | 8 |
| RC4_40 | Stream | 5 | 16 | 0 | N/A |
| RC4_128 | Stream | 16 | 16 | 0 | N/A |
| DES40_CBC | Block | 5 | 8 | 8 | 8 |
| DES_CBC | Block | 8 | 8 | 8 | 8 |
| 3DES_EDE_CBC | Block | 24 | 24 | 8 | 8 |

Type
: Indicates whether this is a stream cipher or a block cipher running in CBC mode.

Key Material
: The number of bytes from the key_block that are used for generating the write keys.

Expanded Key Material
: The number of bytes actually fed into the encryption algorithm.

IV Size
: The amount of data needed to be generated for the initialization vector. Zero for stream ciphers; equal to the block size for block ciphers.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

Block Size
The amount of data a block cipher enciphers in one chunk; a block cipher running in CBC mode can only encrypt an even multiple of its block size.

| Hash function | Hash Size | Padding Size |
|---|---|---|
| NULL | 0 | 0 |
| MD5 | 16 | 48 |
| SHA | 20 | 40 |

## Appendix D. Implementation Notes

The TLS protocol cannot prevent many common security mistakes. This section provides several recommendations to assist implementors.




### D.1. Random Number Generation and Seeding

TLS requires a cryptographically secure pseudorandom number generator (PRNG). Care must be taken in designing and seeding PRNGs. PRNGs based on secure hash operations, most notably MD5 and/or SHA, are acceptable, but cannot provide more security than the size of the random number generator state. (For example, MD5-based PRNGs usually provide 128 bits of state.)

To estimate the amount of seed material being produced, add the number of bits of unpredictable information in each seed byte. For example, keystroke timing values taken from a PC compatible's 18.2 Hz timer provide 1 or 2 secure bits each, even though the total size of the counter value is 16 bits or more. Seeding a 128-bit PRNG would thus require approximately 100 such timer values.

[RANDOM] provides guidance on the generation of random values.

### D.2 Certificates and Authentication

Implementations are responsible for verifying the integrity of certificates and should generally support certificate revocation messages. Certificates should always be verified to ensure proper signing by a trusted Certificate Authority (CA). The selection and addition of trusted CAs should be done very carefully. Users should be able to view information about the certificate and root CA.

### D.3 CipherSuites

TLS supports a range of key sizes and security levels, including some that provide no or minimal security. A proper implementation will probably not support many cipher suites. For example, 40-bit encryption is easily broken, so implementations requiring strong security should not allow 40-bit keys. Similarly, anonymous Diffie-Hellman is strongly discouraged because it cannot prevent man-in-

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

the-middle attacks. Applications should also enforce minimum and maximum key sizes. For example, certificate chains containing 512-bit RSA keys or signatures are not appropriate for high-security applications.





## Appendix E. Backward Compatibility with SSL

For historical reasons and in order to avoid a profligate consumption of reserved port numbers, application protocols that are secured by TLS 1.1, TLS 1.0, SSL 3.0, and SSL 2.0 all frequently share the same connection port. For example, the https protocol (HTTP secured by SSL or TLS) uses port 443 regardless of which security protocol it is using. Thus, some mechanism must be determined to distinguish and negotiate among the various protocols.

TLS versions 1.1 and 1.0, and SSL 3.0 are very similar; thus, supporting both is easy. TLS clients who wish to negotiate with such older servers SHOULD send client hello messages using the SSL 3.0 record format and client hello structure, sending {3, 2} for the version field to note that they support TLS 1.1. If the server supports only TLS 1.0 or SSL 3.0, it will respond with a downrev 3.0 server hello; if it supports TLS 1.1 it will respond with a TLS 1.1 server hello. The negotiation then proceeds as appropriate for the negotiated protocol.

Similarly, a TLS 1.1 server that wishes to interoperate with TLS 1.0 or SSL 3.0 clients SHOULD accept SSL 3.0 client hello messages and respond with a SSL 3.0 server hello if an SSL 3.0 client hello with a version field of {3, 0} is received, denoting that this client does not support TLS. Similarly, if a SSL 3.0 or TLS 1.0 hello with a version field of {3, 1} is received, the server SHOULD respond with a TLS 1.0 hello with a version field of {3, 1}.

Whenever a client already knows the highest protocol known to a server (for example, when resuming a session), it SHOULD initiate the connection in that native protocol.

TLS 1.1 clients that support SSL Version 2.0 servers MUST send SSL Version 2.0 client hello messages [SSL2]. TLS servers SHOULD accept either client hello format if they wish to support SSL 2.0 clients on the same connection port. The only deviations from the Version 2.0 specification are the ability to specify a version with a value of three and the support for more ciphering types in the CipherSpec.

Warning: The ability to send Version 2.0 client hello messages will be phased out with all due haste. Implementors SHOULD make every effort to move forward as quickly as possible. Version 3.0

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

provides better mechanisms for moving to newer versions.



The following cipher specifications are carryovers from SSL Version 2.0. These are assumed to use RSA for key exchange and authentication.

```
 V2CipherSpec TLS_RC4_128_WITH_MD5          = { 0x01,0x00,0x80 };
 V2CipherSpec TLS_RC4_128_EXPORT40_WITH_MD5 = { 0x02,0x00,0x80 };
 V2CipherSpec TLS_RC2_CBC_128_CBC_WITH_MD5  = { 0x03,0x00,0x80 };
 V2CipherSpec TLS_RC2_CBC_128_CBC_EXPORT40_WITH_MD5
                                            = { 0x04,0x00,0x80 };
 V2CipherSpec TLS_IDEA_128_CBC_WITH_MD5     = { 0x05,0x00,0x80 };
 V2CipherSpec TLS_DES_64_CBC_WITH_MD5       = { 0x06,0x00,0x40 };
 V2CipherSpec TLS_DES_192_EDE3_CBC_WITH_MD5 = { 0x07,0x00,0xC0 };
```

Cipher specifications native to TLS can be included in Version 2.0 client hello messages using the syntax below. Any V2CipherSpec element with its first byte equal to zero will be ignored by Version 2.0 servers. Clients sending any of the above V2CipherSpecs SHOULD also include the TLS equivalent (see Appendix A.5):

```
 V2CipherSpec (see TLS name) = { 0x00, CipherSuite };
```

Note: TLS 1.1 clients may generate the SSLv2 EXPORT cipher suites in handshakes for backward compatibility but MUST NOT negotiate them in TLS 1.1 mode.

### E.1. Version 2 Client Hello

The Version 2.0 client hello message is presented below using this document's presentation model. The true definition is still assumed to be the SSL Version 2.0 specification. Note that this message MUST be sent directly on the wire, not wrapped as an SSLv3 record

```
  uint8 V2CipherSpec[3];

  struct {
      uint16 msg_length;
      uint8 msg_type;
      Version version;
      uint16 cipher_spec_length;
      uint16 session_id_length;
      uint16 challenge_length;
      V2CipherSpec cipher_specs[V2ClientHello.cipher_spec_length];
      opaque session_id[V2ClientHello.session_id_length];
      opaque challenge[V2ClientHello.challenge_length;
  } V2ClientHello;
```

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

msg_length
This field is the length of the following data in bytes. The high bit MUST be 1 and is not part of the length.

msg_type
This field, in conjunction with the version field, identifies a version 2 client hello message. The value SHOULD be one (1).

version
The highest version of the protocol supported by the client (equals ProtocolVersion.version; see Appendix A.1).

cipher_spec_length
This field is the total length of the field cipher_specs. It cannot be zero and MUST be a multiple of the V2CipherSpec length (3).

session_id_length
This field MUST have a value of zero.

challenge_length
The length in bytes of the client's challenge to the server to authenticate itself. When using the SSLv2 backward compatible handshake the client MUST use a 32-byte challenge.

cipher_specs
This is a list of all CipherSpecs the client is willing and able to use. There MUST be at least one CipherSpec acceptable to the server.

session_id
This field MUST be empty.

challenge The client challenge to the server for the server to identify itself is a (nearly) arbitrary-length random. The TLS server will right-justify the challenge data to become the ClientHello.random data (padded with leading zeroes, if necessary), as specified in this protocol specification. If the length of the challenge is greater than 32 bytes, only the last 32 bytes are used. It is legitimate (but not necessary) for a V3 server to reject a V2 ClientHello that has fewer than 16 bytes of challenge data.

Note: Requests to resume a TLS session MUST use a TLS client hello.



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

### E.2. Avoiding Man-in-the-Middle Version Rollback

When TLS clients fall back to Version 2.0 compatibility mode, they SHOULD use special PKCS #1 block formatting. This is done so that TLS servers will reject Version 2.0 sessions with TLS-capable clients.

When TLS clients are in Version 2.0 compatibility mode, they set the right-hand (least significant) 8 random bytes of the PKCS padding (not including the terminal null of the padding) for the RSA encryption of the ENCRYPTED-KEY-DATA field of the CLIENT-MASTER-KEY to 0x03 (the other padding bytes are random). After decrypting the ENCRYPTED-KEY-DATA field, servers that support TLS SHOULD issue an error if these eight padding bytes are 0x03. Version 2.0 servers receiving blocks padded in this manner will proceed normally.

## Appendix F. Security Analysis

The TLS protocol is designed to establish a secure connection between a client and a server communicating over an insecure channel. This document makes several traditional assumptions, including that attackers have substantial computational resources and cannot obtain secret information from sources outside the protocol. Attackers are assumed to have the ability to capture, modify, delete, replay, and otherwise tamper with messages sent over the communication channel. This appendix outlines how TLS has been designed to resist a variety of attacks.

### F.1. Handshake Protocol

The handshake protocol is responsible for selecting a CipherSpec and generating a Master Secret, which together comprise the primary cryptographic parameters associated with a secure session. The handshake protocol can also optionally authenticate parties who have certificates signed by a trusted certificate authority.

#### F.1.1. Authentication and Key Exchange

TLS supports three authentication modes: authentication of both parties, server authentication with an unauthenticated client, and total anonymity. Whenever the server is authenticated, the channel is secure against man-in-the-middle attacks, but completely anonymous sessions are inherently vulnerable to such attacks. Anonymous servers cannot authenticate clients. If the server is authenticated, its certificate message must provide a valid certificate chain leading to an acceptable certificate authority. Similarly, authenticated clients must supply an acceptable certificate to the





server. Each party is responsible for verifying that the other's certificate is valid and has not expired or been revoked.

The general goal of the key exchange process is to create a pre_master_secret known to the communicating parties and not to attackers. The pre_master_secret will be used to generate the master_secret (see Section 8.1). The master_secret is required to

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

generate the finished messages, encryption keys, and MAC secrets (see Sections 7.4.8, 7.4.9, and 6.3). By sending a correct finished message, parties thus prove that they know the correct pre_master_secret.

F.1.1.1. Anonymous Key Exchange

Completely anonymous sessions can be established using RSA or Diffie-Hellman for key exchange. With anonymous RSA, the client encrypts a pre_master_secret with the server's uncertified public key extracted from the server key exchange message. The result is sent in a client key exchange message. Since eavesdroppers do not know the server's private key, it will be infeasible for them to decode the pre_master_secret.

Note: No anonymous RSA Cipher Suites are defined in this document.

With Diffie-Hellman, the server's public parameters are contained in the server key exchange message and the client's are sent in the client key exchange message. Eavesdroppers who do not know the private values should not be able to find the Diffie-Hellman result (i.e., the pre_master_secret).

Warning: Completely anonymous connections only provide protection against passive eavesdropping. Unless an independent tamper-proof channel is used to verify that the finished messages were not replaced by an attacker, server authentication is required in environments where active man-in-the-middle attacks are a concern.

F.1.1.2. RSA Key Exchange and Authentication

With RSA, key exchange and server authentication are combined. The public key either may be contained in the server's certificate or may be a temporary RSA key sent in a server key exchange message. When temporary RSA keys are used, they are signed by the server's RSA certificate. The signature includes the current ClientHello.random, so old signatures and temporary keys cannot be replayed. Servers may use a single temporary RSA key for multiple negotiation sessions.

Note: The temporary RSA key option is useful if servers need large





certificates but must comply with government-imposed size limits on keys used for key exchange.

Note that if ephemeral RSA is not used, compromise of the server's static RSA key results in a loss of confidentiality for all sessions protected under that static key. TLS users desiring Perfect Forward Secrecy should use DHE cipher suites. The damage done by exposure of a private key can be limited by changing one's private key (and certificate) frequently.

After verifying the server's certificate, the client encrypts a pre_master_secret with the server's public key. By successfully decoding the pre_master_secret and producing a correct finished message, the server demonstrates that it knows the private key

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1023
Keysight, EX-1045
Page 1023

corresponding to the server certificate.

When RSA is used for key exchange, clients are authenticated using the certificate verify message (see Section 7.4.8). The client signs a value derived from the master_secret and all preceding handshake messages. These handshake messages include the server certificate, which binds the signature to the server, and ServerHello.random, which binds the signature to the current handshake process.

F.1.1.3. Diffie-Hellman Key Exchange with Authentication

When Diffie-Hellman key exchange is used, the server can either supply a certificate containing fixed Diffie-Hellman parameters or use the server key exchange message to send a set of temporary Diffie-Hellman parameters signed with a DSS or RSA certificate. Temporary parameters are hashed with the hello.random values before signing to ensure that attackers do not replay old parameters. In either case, the client can verify the certificate or signature to ensure that the parameters belong to the server.

If the client has a certificate containing fixed Diffie-Hellman parameters, its certificate contains the information required to complete the key exchange. Note that in this case the client and server will generate the same Diffie-Hellman result (i.e., pre_master_secret) every time they communicate. To prevent the pre_master_secret from staying in memory any longer than necessary, it should be converted into the master_secret as soon as possible. Client Diffie-Hellman parameters must be compatible with those supplied by the server for the key exchange to work.

If the client has a standard DSS or RSA certificate or is unauthenticated, it sends a set of temporary parameters to the server in the client key exchange message, then optionally uses a certificate verify message to authenticate itself.





If the same DH keypair is to be used for multiple handshakes, either because the client or server has a certificate containing a fixed DH keypair or because the server is reusing DH keys, care must be taken to prevent small subgroup attacks. Implementations SHOULD follow the guidelines found in [SUBGROUP].

Small subgroup attacks are most easily avoided by using one of the DHE ciphersuites and generating a fresh DH private key (X) for each handshake. If a suitable base (such as 2) is chosen, g^X mod p can be computed very quickly, therefore the performance cost is minimized. Additionally, using a fresh key for each handshake provides Perfect Forward Secrecy. Implementations SHOULD generate a new X for each handshake when using DHE ciphersuites.

F.1.2. Version Rollback Attacks

Because TLS includes substantial improvements over SSL Version 2.0, attackers may try to make TLS-capable clients and servers fall back to Version 2.0. This attack can occur if (and only if) two TLS-capable parties use an SSL 2.0 handshake.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

Although the solution using non-random PKCS #1 block type 2 message padding is inelegant, it provides a reasonably secure way for Version 3.0 servers to detect the attack. This solution is not secure against attackers who can brute force the key and substitute a new ENCRYPTED-KEY-DATA message containing the same key (but with normal padding) before the application specified wait threshold has expired. Parties concerned about attacks of this scale should not use 40-bit encryption keys. Altering the padding of the least-significant 8 bytes of the PKCS padding does not impact security for the size of the signed hashes and RSA key lengths used in the protocol, since this is essentially equivalent to increasing the input block size by 8 bytes.

### F.1.3. Detecting Attacks against the Handshake Protocol

An attacker might try to influence the handshake exchange to make the parties select different encryption algorithms than they would normally chooses.

For this attack, an attacker must actively change one or more handshake messages. If this occurs, the client and server will compute different values for the handshake message hashes. As a result, the parties will not accept each others' finished messages. Without the master_secret, the attacker cannot repair the finished messages, so the attack will be discovered.





### F.1.4. Resuming Sessions

When a connection is established by resuming a session, new ClientHello.random and ServerHello.random values are hashed with the session's master_secret. Provided that the master_secret has not been compromised and that the secure hash operations used to produce the encryption keys and MAC secrets are secure, the connection should be secure and effectively independent from previous connections. Attackers cannot use known encryption keys or MAC secrets to compromise the master_secret without breaking the secure hash operations (which use both SHA and MD5).

Sessions cannot be resumed unless both the client and server agree. If either party suspects that the session may have been compromised, or that certificates may have expired or been revoked, it should force a full handshake. An upper limit of 24 hours is suggested for session ID lifetimes, since an attacker who obtains a master_secret may be able to impersonate the compromised party until the corresponding session ID is retired. Applications that may be run in relatively insecure environments should not write session IDs to stable storage.

### F.1.5. MD5 and SHA

TLS uses hash functions very conservatively. Where possible, both MD5 and SHA are used in tandem to ensure that non-catastrophic flaws in one algorithm will not break the overall protocol.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

F.2. Protecting Application Data

The master_secret is hashed with the ClientHello.random and ServerHello.random to produce unique data encryption keys and MAC secrets for each connection.

Outgoing data is protected with a MAC before transmission. To prevent message replay or modification attacks, the MAC is computed from the MAC secret, the sequence number, the message length, the message contents, and two fixed character strings. The message type field is necessary to ensure that messages intended for one TLS Record Layer client are not redirected to another. The sequence number ensures that attempts to delete or reorder messages will be detected. Since sequence numbers are 64 bits long, they should never overflow. Messages from one party cannot be inserted into the other's output, since they use independent MAC secrets. Similarly, the server-write and client-write keys are independent, so stream cipher keys are used only once.





If an attacker does break an encryption key, all messages encrypted with it can be read. Similarly, compromise of a MAC key can make message modification attacks possible. Because MACs are also encrypted, message-alteration attacks generally require breaking the encryption algorithm as well as the MAC.

Note: MAC secrets may be larger than encryption keys, so messages can remain tamper resistant even if encryption keys are broken.

F.3. Explicit IVs

[CBCATT] describes a chosen plaintext attack on TLS that depends on knowing the IV for a record. Previous versions of TLS [TLS1.0] used the CBC residue of the previous record as the IV and therefore enabled this attack. This version uses an explicit IV in order to protect against this attack.

F.4. Security of Composite Cipher Modes

TLS secures transmitted application data via the use of symmetric encryption and authentication functions defined in the negotiated ciphersuite. The objective is to protect both the integrity and confidentiality of the transmitted data from malicious actions by active attackers in the network. It turns out that the order in which encryption and authentication functions are applied to the data plays an important role for achieving this goal [ENCAUTH].

The most robust method, called encrypt-then-authenticate, first applies encryption to the data and then applies a MAC to the ciphertext. This method ensures that the integrity and confidentiality goals are obtained with ANY pair of encryption and MAC functions, provided that the former is secure against chosen plaintext attacks and that the MAC is secure against chosen-message attacks. TLS uses another method, called authenticate-then-encrypt, in which first a MAC is computed on the plaintext and then the

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1026
Keysight, EX-1045
Page 1026

concatenation of plaintext and MAC is encrypted. This method has been proven secure for CERTAIN combinations of encryption functions and MAC functions, but it is not guaranteed to be secure in general. In particular, it has been shown that there exist perfectly secure encryption functions (secure even in the information-theoretic sense) that combined with any secure MAC function, fail to provide the confidentiality goal against an active attack. Therefore, new ciphersuites and operation modes adopted into TLS need to be analyzed under the authenticate-then-encrypt method to verify that they achieve the stated integrity and confidentiality goals.





Currently, the security of the authenticate-then-encrypt method has been proven for some important cases. One is the case of stream ciphers in which a computationally unpredictable pad of the length of the message, plus the length of the MAC tag, is produced using a pseudo-random generator and this pad is xor-ed with the concatenation of plaintext and MAC tag. The other is the case of CBC mode using a secure block cipher. In this case, security can be shown if one applies one CBC encryption pass to the concatenation of plaintext and MAC and uses a new, independent, and unpredictable IV for each new pair of plaintext and MAC. In previous versions of SSL, CBC mode was used properly EXCEPT that it used a predictable IV in the form of the last block of the previous ciphertext. This made TLS open to chosen plaintext attacks. This version of the protocol is immune to those attacks. For exact details in the encryption modes proven secure, see [ENCAUTH].

### F.5. Denial of Service

TLS is susceptible to a number of denial of service (DoS) attacks. In particular, an attacker who initiates a large number of TCP connections can cause a server to consume large amounts of CPU doing RSA decryption. However, because TLS is generally used over TCP, it is difficult for the attacker to hide his point of origin if proper TCP SYN randomization is used [SEQNUM] by the TCP stack.

Because TLS runs over TCP, it is also susceptible to a number of denial of service attacks on individual connections. In particular, attackers can forge RSTs, thereby terminating connections, or forge partial TLS records, thereby causing the connection to stall. These attacks cannot in general be defended against by a TCP-using protocol. Implementors or users who are concerned with this class of attack should use IPsec AH [AH-ESP] or ESP [AH-ESP].

### F.6. Final Notes

For TLS to be able to provide a secure connection, both the client and server systems, keys, and applications must be secure. In addition, the implementation must be free of security errors.

The system is only as strong as the weakest key exchange and authentication algorithm supported, and only trustworthy cryptographic functions should be used. Short public keys, 40-bit

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1027
Keysight, EX-1045
Page 1027

bulk encryption keys, and anonymous servers should be used with great caution. Implementations and users must be careful when deciding which certificates and certificate authorities are acceptable; a dishonest certificate authority can do tremendous damage.



Normative References

[AES] National Institute of Standards and Technology, "Specification for the Advanced Encryption Standard (AES)" FIPS 197. November 26, 2001.

[3DES] W. Tuchman, "Hellman Presents No Shortcut Solutions To DES," IEEE Spectrum, v. 16, n. 7, July 1979, pp. 40-41.

[DES] ANSI X3.106, "American National Standard for Information Systems-Data Link Encryption," American National Standards Institute, 1983.

[DSS] NIST FIPS PUB 186-2, "Digital Signature Standard," National Institute of Standards and Technology, U.S. Department of Commerce, 2000.

[HMAC] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[IDEA] X. Lai, "On the Design and Security of Block Ciphers," ETH Series in Information Processing, v. 1, Konstanz: Hartung-Gorre Verlag, 1992.

[MD5] Rivest, R., "The MD5 Message-Digest Algorithm ", RFC 1321, April 1992.

[PKCS1A] B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 1.5", RFC 2313, March 1998.

[PKCS1B] J. Jonsson, B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, February 2003.

[PKIX] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.

[RC2] Rivest, R., "A Description of the RC2(r) Encryption Algorithm", RFC 2268, March 1998.

[SCH] B. Schneier. "Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2ed", Published by John Wiley & Sons, Inc. 1996.

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

[SHA] NIST FIPS PUB 180-2, "Secure Hash Standard," National Institute of Standards and Technology, U.S. Department of Commerce., August 2001.

[REQ] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[TLSAES] Chown, P., "Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)", RFC 3268, June 2002.

[TLSEXT] Blake-Wilson, S., Nystrom, M., Hopwood, D., Mikkelsen, J., and T. Wright, "Transport Layer Security (TLS) Extensions", RFC 3546, June 2003.

[TLSKRB] Medvinsky, A. and M. Hur, "Addition of Kerberos Cipher Suites to Transport Layer Security (TLS)", RFC 2712, October 1999.

Informative References

[AH-ESP] Kent, S., "IP Authentication Header", RFC 4302, December 2005.

Eastlake 3rd, D., "Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 4305, December 2005.

[BLEI] Bleichenbacher D., "Chosen Ciphertext Attacks against Protocols Based on RSA Encryption Standard PKCS #1" in Advances in Cryptology -- CRYPTO'98, LNCS vol. 1462, pages: 1-12, 1998.

[CBCATT] Moeller, B., "Security of CBC Ciphersuites in SSL/TLS: Problems and Countermeasures", http://www.openssl.org/~bodo/tls-cbc.txt.

[CBCTIME] Canvel, B., "Password Interception in a SSL/TLS Channel", http://lasecwww.epfl.ch/memo_ssl.shtml, 2003.

[ENCAUTH] Krawczyk, H., "The Order of Encryption and Authentication for Protecting Communications (Or: How Secure is SSL?)", Crypto 2001.



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1029
Keysight, EX-1045
Page 1029

[KPR03] Klima, V., Pokorny, O., Rosa, T., "Attacking RSA-based Sessions in SSL/TLS", http://eprint.iacr.org/2003/052/, March 2003.

[PKCS6] RSA Laboratories, "PKCS #6: RSA Extended Certificate Syntax Standard," version 1.5, November 1993.

[PKCS7] RSA Laboratories, "PKCS #7: RSA Cryptographic Message Syntax Standard," version 1.5, November 1993.

[RANDOM] Eastlake, D., 3rd, Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.

[RSA] R. Rivest, A. Shamir, and L. M. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," Communications of the ACM, v. 21, n. 2, Feb 1978, pp. 120-126.

[SEQNUM] Bellovin, S., "Defending Against Sequence Number Attacks", RFC 1948, May 1996.

[SSL2] Hickman, Kipp, "The SSL Protocol", Netscape Communications Corp., Feb 9, 1995.

[SSL3] A. Frier, P. Karlton, and P. Kocher, "The SSL 3.0 Protocol", Netscape Communications Corp., Nov 18, 1996.

[SUBGROUP] Zuccherato, R., "Methods for Avoiding the "Small-Subgroup" Attacks on the Diffie-Hellman Key Agreement Method for S/MIME", RFC 2785, March 2000.

[TCP] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, June 2005.

[TIMING] Boneh, D., Brumley, D., "Remote timing attacks are practical", USENIX Security Symposium 2003.

[TLS1.0] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[X501] ITU-T Recommendation X.501: Information Technology - Open Systems Interconnection - The Directory: Models, 1993.

[X509] ITU-T Recommendation X.509 (1997 E): Information Technology - Open Systems Interconnection - "The Directory - Authentication Framework". 1988.





[XDR] Srinivasan, R., "XDR: External Data Representation Standard", RFC 1832, August 1995.

Authors' Addresses

Working Group Chairs

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1030
Keysight, EX-1045
Page 1030

Win Treese

EMail: treese@acm.org

Eric Rescorla

EMail: ekr@rtfm.com

Editors

Tim Dierks
Independent

EMail: tim@dierks.org

Eric Rescorla
RTFM, Inc.

EMail: ekr@rtfm.com

Other Contributors

Christopher Allen (co-editor of TLS 1.0)
Alacrity Ventures
EMail: ChristopherA@AlacrityManagement.com

Martin Abadi
University of California, Santa Cruz
EMail: abadi@cs.ucsc.edu

Ran Canetti
IBM
EMail: canetti@watson.ibm.com





Taher Elgamal
Securify
EMail: taher@securify.com

Anil Gangolli
EMail: anil@busybuddha.org

Kipp Hickman

Phil Karlton (co-author of SSLv3)

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1031
Keysight, EX-1045
Page 1031

Paul Kocher (co-author of SSLv3)
Cryptography Research
EMail: paul@cryptography.com

Hugo Krawczyk
Technion Israel Institute of Technology
EMail: hugo@ee.technion.ac.il

Robert Relyea
Netscape Communications
EMail: relyea@netscape.com

Jim Roskind
Netscape Communications
EMail: jar@netscape.com

Michael Sabin

Dan Simon
Microsoft, Inc.
EMail: dansimon@microsoft.com

Tom Weinstein



Comments

The discussion list for the IETF TLS working group is located at the e-mail address <ietf-tls@lists.consensus.com>. Information on the group and information on how to subscribe to the list is at <http://lists.consensus.com/>.

Archives of the list can be found at:
<http://www.imc.org/ietf-tls/mail-archive/>

https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1033
Keysight, EX-1045
Page 1033

assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).



https://web.archive.org/web/20060626134135if_/http://www.ietf.org:80/rfc/rfc4346.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1034
Keysight, EX-1045
Page 1034

https://web.archive.org/web/20060902171316/https://www.cs.unm.edu/~treport/tr/02-12/firewall.pdf

# A History and Survey of Network Firewalls

KENNETH INGHAM
Kenneth Ingham Consulting
and
STEPHANIE FORREST
University of New Mexico

Firewalls are network devices which enforce an organization's security policy. Since their development, various methods have been used to implement firewalls. These methods filter network traffic at one or more of the seven layers of the ISO network model, most commonly at the application, transport, and network, and data-link levels. In addition, researchers have developed some newer methods, such as protocol normalization and distributed firewalls, which have not yet been widely adopted.

Firewalls involve more than the technology to implement them. Specifying a set of filtering rules, known as a *policy*, is typically complicated and error-prone. High-level languages have been developed to simplify the task of correctly defining a firewall's policy. Once a policy has been specified, the firewall needs to be tested to determine if it actually implements the policy correctly. Little work exists in the area of firewall theory; however, this article summarizes what exists.

Because some data must be able to pass in and out of a firewall, in order for the protected network to be useful, not all attacks can be stopped by firewalls. Some emerging technologies, such as Virtual Private Networks (VPN) and peer-to-peer networking pose new challenges for firewalls.

Categories and Subject Descriptors: C.2.0 [**COMPUTER-COMMUNICATION NETWORKS**]: General

General Terms: security

Additional Key Words and Phrases: Firewalls, Network Security

The University of New Mexico Computer Science Department Technical Report 2002-37.
Author's addresses: K. Ingham, Kenneth Ingham Consulting, 1601 Rita Dr NE, Albuquerque, NM 87106-1127, ingham@i-pi.com. S. Forrest, Department of Computer Science, University of New Mexico, Albuquerque, NM 87131, forrest@cs.unm.edu.
Permission to make digital/hard copy of all or part of this material without fee for personal or classroom use provided that the copies are not made or distributed for profit or commercial advantage, the ACM copyright/server notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee.
© 20YY ACM 0000-0000/20YY/0000-0001 $5.00



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1036

**Keysight, EX-1045**
**Page 1036**

## 1. INTRODUCTION

The idea of a wall to keep out intruders dates back thousands of years. For example, over two thousand years ago, the Chinese built the Great Wall as protection from neighboring northern tribes. A second example is that of European kings who built castles with high walls and moats to protect themselves and their subjects, both from invading armies and from marauding bands intent on pillaging and looting. The term "firewall" was in use by Lightoler as early as [1764] to describe walls which separated the parts of a building most likely to have a fire (e.g., a kitchen) from the rest of a structure. These physical barriers prevented or slowed a fire's spread throughout a building, saving both lives and property. A related use of the term arose in connection with steam trains, as described by Schneier [2000]:

> Coal-powered trains had a large furnace in the engine room, along with a pile of coal. The engineer would shovel coal into the engine. This process created coal dust, which was highly flammable. Occasionally the coal dust would catch fire, causing an engine fire that sometimes spread into the passenger cars. Since dead passengers reduced revenue, train engines were built with iron walls right behind the engine compartment. This stopped fires from spreading into the passenger cars, but didn't protect the engineer between the coal pile and the furnace.

In this paper we will be concerned with firewalls in a more modern setting—computer networks. The predecessors to firewalls for network security were the routers used in the late 1980s to separate networks from one another. A network misconfiguration which caused problems on one side of the router was largely isolated from the network on the other side. In a similar vein, so-called "chatty" protocols on one network (which used broadcasts for much of their configuration) would not affect the other network's bandwidth [Avolio 1999; Schneier 2000]. From these historical examples we can see how the term "firewall" came to describe a device or collection of devices which separates its occupants from potentially dangerous external environments (e.g., the Internet). A firewall is designed to prevent or slow the spread of dangerous events.

For the purposes of this paper, we define a firewall as a machine or collection of machines between two networks, meeting the following criteria:

—The firewall is at the boundary between the two networks;

—All traffic between the two networks must pass through the firewall;

—The firewall has a mechanism to allow some traffic to pass while blocking other traffic. The rules describing what traffic is allowed enforce the firewall's *policy*.

Additional desirable criteria include:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1037

Keysight, EX-1045
Page 1037

—Resistance to security compromise;

—Auditing and accounting capabilities;

—Resource monitoring;

—No user accounts or direct user access;

—Strong authentication for proxies (e.g., smart cards rather than simple passwords);

—Fail-safety. If it fails, the protected system(s) is(are) still secure because no traffic is allowed to pass through the firewall.

Our review of firewall technologies proceeds roughly in historical order. Interestingly, the historical development of firewalls parallels the descending levels of the protocol stack:

| **ISO level** | **Internet example** |
|---|---|
| Application | File Transfer Protocol (FTP), Telnet |
| Presentation | e.g., Common Object Request Broker Architecture (CORBA) |
| Session | *no directly corresponding protocol* |
| Transport | Transport Control Protocol (TCP), User Datagram Protocol (UDP) |
| Network | Internet Protocol (IP) |
| Data link | Ethernet or Asynchronous Transfer Mode (ATM) |
| Physical | twisted pair or fiber optic cable |

The protocols used on the Internet for these layers, as well as all other Internet standards are specified by documents known as Requests for Comments (RFCs). RFCs often provide information beyond the bare specifications of the standard, and can be useful network administrators, and they appear frequently as citations in this article. For more information about RFCs, read the RFC frequently-asked question list [The RFC Editor 2001].

Following the order of much of the historical development of firewalls, this paper is organized around the levels of the protocol stack, describing approaches which filter or are otherwise directed at each level. Section 2 sets the stage by describing the many reasons why organizations and individuals need firewalls. The next section, Section 3, describes other surveys of firewall technologies. Sections 4-7 descend through the protocol stack, discussing firewall technologies at each of the four relevant levels. Figure 1 provides a graphical view of several of the architectures that we cover in remainder of the paper. Many of the firewalls we discuss use or have used proxies as a major part of how they protect networks. A proxy is a program that receives the traffic destined for another computer. Proxies sometimes require user authentication; they then verify that the user is allowed to connect to



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1038

Keysight, EX-1045
Page 1038

the destination, and then they connect to the destination service on behalf of the user.

Section 8 considers firewall designs which do not fit cleanly into one of the various network protocol layers, for example, methods addressing multicast services, distributed firewalls, and protocol normalization. Firewalls are a large part of the commercial network security market. Many companies market products which filter network traffic one of the ISO levels. Although not the primary focus of this paper, in Section 9 we consider important examples of these commercial products and show how they fit into the existing literature. Specifying firewall policies can be a complex task, and consequently, some higher-level languages have been developed in which to express policies (Section 10). Once a policy has been specified, the next question is, "How can it be determined that a given firewall correctly implements the policy?" Firewall testing is one way to address this question, Section 11 reviews the literature on this topic. Although little theory has been developed about firewalls, Section 12 describes the existing theoretical literature.

Very few computer networks can afford to be completely isolated from external networks. Thus, an important job of a modern firewall is controlling what traffic is allowed through the firewall. Detecting hostile data in this traffic is an important problem confronted by modern firewalls, one that we discuss in Section 13. This discussion leads to the related topics of expected future challenges for firewalls (Section 14) and some speculations about how firewalls have influenced recent protocols and how they are likely to change in the future (Section 15).

## 2. THE NEED FOR FIREWALLS

In the early years, the Internet supported a relatively small community of compatible users who valued openness for sharing and collaboration. This view was challenged by the Morris Worm [Spafford 1988; Eichin and Rochlis 1989; Denning 1989; Spafford 1991]. However, even without the Morris worm, the end of the open, trusting community would have come soon through growth and diversification. Examples of successful or attempted intrusions around the same time include: Clifford Stoll's discovery of German spies tampering with his system [Stoll 1988; 1989], and Bill Cheswick's "Evening with Berferd" [1992] in which he set up a simple electronic "jail" for an attacker. In this jail, the attacker was unable to affect the real system but was left with the impression that he or she had successfully broken in. Cheswick was able to observe everything the attacker did, learning from these actions, and alerting system administrators of the networks from where the attacks were originating. Such incidents clearly signaled the end of an open and benign Internet. By [1992] Steve Bellovin described a collection of attacks that he had noticed while monitoring the AT&T firewall and the networks around it. The



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1039

Keysight, EX-1045
Page 1039

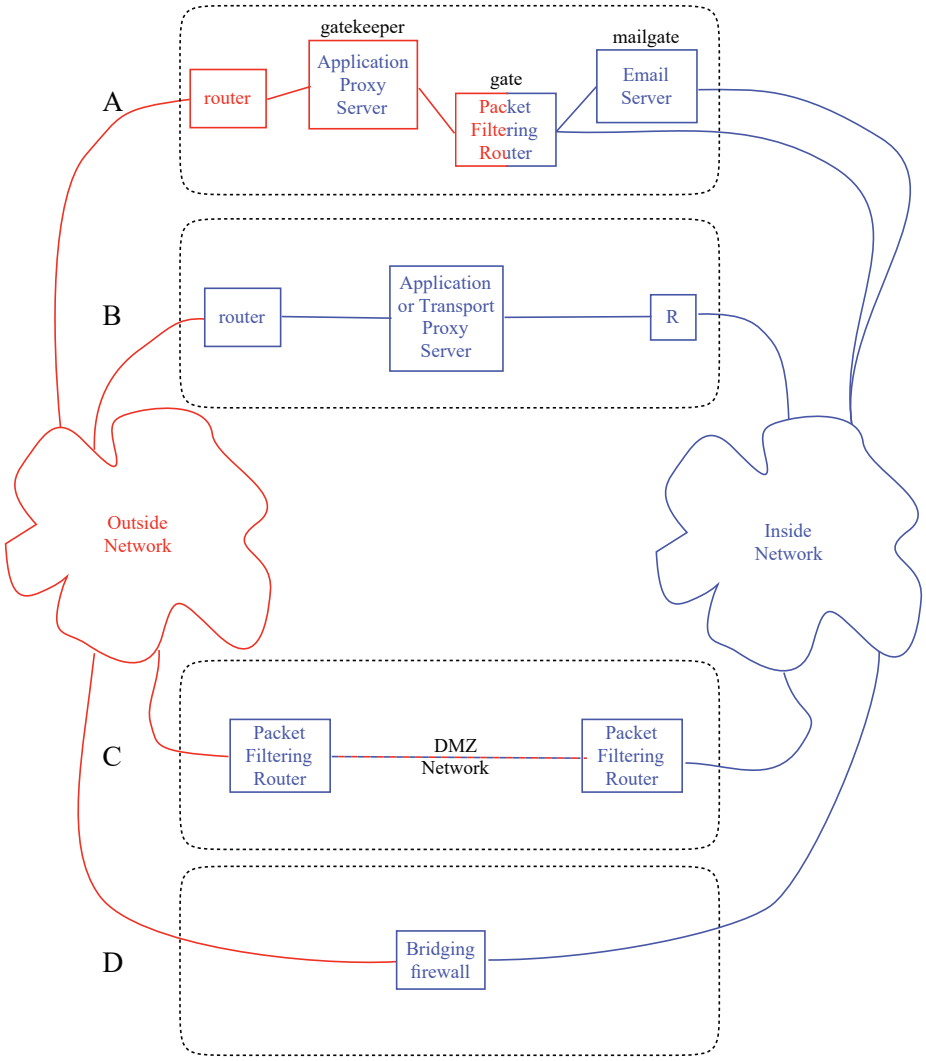



Fig. 1. Common firewall architectures, referred to throughout this paper. The blue portion of the picture represents the trusted parts of the network, and the red portion indicates machines or networks which are not trusted; these colors came from the original DEC documentation for their firewall described in Section 4.1. A) The DEC SEAL from Section 4.1. B) A firewall using only application- or transport-level proxies, a more common architecture than the DEC SEAL architecture due to the reduced cost of hardware. The box containing just the "R" is an optional router in this architecture. This architecture with the transport-level proxy server is similar to the AT&T firewall described in Section 5.1. C) A firewall using two packet filtering routers (described in Section 6.1) and illustrating one way of creating a DMZ, sometimes also known as a screened network. A DMZ network consists of a network of machines offering services to the Internet. If a machine here is compromised, the inside network remains safe. D) A bridging firewall similar to that described in Section 7.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1040

Keysight, EX-1045
Page 1040

result was clear—there were many untrustworthy and even malicious users on the Internet.

Because not everybody can be trusted, when networks are connected together, a different level of trust often exists on the different sides of the connection. "Trust" in this sense means that an organization believes that the both the software and the users on its computers are not malicious. Firewalls can be used to enforce trust boundaries, which are imposed for a variety of reasons:

*Security problems in operating systems:.* Operating systems have a history of insecure configurations. For example, Windows 95 and Windows 98 were widely distributed with windows file sharing turned on by default; a collection of viruses exploited this vulnerability (e.g., [Computer Emergency Response Team (CERT) 2000b; 2000c]). A second example is Red Hat Linux versions 6.2 and 7.0, which were vulnerable to three remote exploits when installed using default options [Computer Emergency Response Team (CERT) 2001a]. It is an on-going and expensive process to secure every user's machine, and many organizations make a conscious decision not to secure the machines inside their firewall. If a machine on the inside is ever compromised, the remaining machines are likely as vulnerable [Muffett 1995], a situation that has been described by Cheswick as "a sort of crunchy shell around a soft, chewy center" [1990].

Individuals can protect a single machine connected to the Internet by purchasing a personal firewall. Rather than trying to secure the underlying operating system, these firewalls simply prevent some types of communication. Such firewalls are often used in homes and on laptops when they are outside their normal firewall. In this case, the trust boundary is the network interface of the machine.

*Preventing access to information:.* A second example of protecting a network is the use of national firewalls, for example, China [McKay 1998]. This firewall exists not to protect them from attack, but instead to (attempt to) limit the activities of their users on the Internet. A similar idea is the use of filtering mandated in the US by the Children's Internet Protection Act (CHIPA). This law requires that schools and libraries which receive federal funding put certain filters on all web content.

*Preventing Information Leaks.* Because all traffic leaving a network must pass through the firewall, it can be used to reduce information leaks, as in [Ranum 1992]:

> The key criterion for success for the Digital corporate gateways is preventing an unauthorized or unnoticed leak of data to the outside.

*Enforcing Policy.* Firewalls are one part of an overall security policy; they enforce the policy of network traffic allowed to enter or leave a network. These policies may limit the applications used, remote machines which may be contacted, and/or the



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1041

Keysight, EX-1045
Page 1041

bandwidth allowed [Treese and Wolman 1993; Hambridge and Sedayao 1993].

*Auditing.* If a security breach (which does not include the firewall) occurs, audit trails can be used to help determine what happened. Audit trails have also been used to take employment actions against employees using work network resources for non-work purposes.

## 3. OTHER SURVEYS

Firewalls have existed since about 1987, and several surveys and histories have already been written. However, none of them provide both the depth and breadth of this survey, nor do they focus on the peer-reviewed literature describing firewall technology.

In [1994], Alec Muffett wrote a paper which provided an excellent review of the firewall policies and architectures of the time. This paper was aimed at people considering implementing a firewall, describing the technologies which they might select, their tradeoffs, and how to maintain a firewall.

One section of the Internet standards document RFC 1636 [Braden et al. 1994] is about the status of firewalls as of February, 1994. In this section, they discuss the problem of false security that a firewall often provides to an organization behind one. They also review the concepts of application- and transport-level proxies, as well as simple packet filtering. They discuss the problems of authentication and enforcing policy, and provide some solutions from the time. One of the most important parts of the firewall section is a discussion of how to design firewall-friendly protocols in the future.

A review of firewalls and their technology appeared in Spectrum [Lodin and Schuba 1998]. This paper is an excellent description of firewalls and their technology at the time it was written. However, it has no references to peer-reviewed literature.

Several books have been written which describe how to build a firewall (e.g., [Cheswick and Bellovin 1994; Zwicky et al. 2000]). These books are excellent for people wanting to either evaluate a commercial firewall or who are implementing their own firewall. However, neither spends much time on firewall history, nor do they provide references to peer-reviewed literature.

In [1997], John Schimmel wrote a historical review of firewall technologies aimed at technical people working in the field of system administration. This review contains good history about early packet filters and a brief overview of proxies. Schimmel also mentions limitations of firewalls, many of which remain to this day and are discussed in this paper in Section 13. Unfortunately, this paper has no references to the original sources of the works described.

A survey of existing firewall technology appeared in Schuba's Ph.D. dissertation, [Schuba 1997]. In this dissertation, Schuba cites many key papers, but he reviews



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1042

Keysight, EX-1045
Page 1042

the literature as it relates to technology, and does not provide as comprehensive a collection of firewall-related references as we do in this paper. Steph: is this too harsh? His work is not bad, and his entire thesis has 148 references. However, his review of firewall literature is only 23 references. His review is really weak compared to this paper.

Also in [1998], Rik Farrow wrote a firewall product analysis which was related to the CSI firewall comparison for that year. This analysis is aimed at management and people just arriving at firewalls, and provides them with the background information they would need to talk with a firewall vendor intelligently.

More recent publications include Frederic Avolio's history of firewalls published in the Cisco Internet Protocol Journal [1999]. Avolio is well-qualified to write such a document, as he was involved with some of the first firewalls. His history describes some of the beginnings of firewalls, from a technical point of view, and aimed at technical people and technically-oriented management. He provides a short description of firewalls which use proxies at the application or transport levels, as well as packet filtering and firewalls which may be a mix of technologies. Rather than providing details, he refers the reader to Cheswick and Bellovin's [1994] book on firewalls. As a contrast with Avolio's history, this paper places emphasis on the academic literature and as a result has substantially more references than Avolio's history.

Habtamu Abie wrote an overview of current firewall technology options and emerging trends in [2000]. He discusses the technology, but does not cite the papers by the people who originally developed this technology. Also, Yakomba Yavwa wrote a similar but less in-depth overview [2000]. Like Abie, Yavwa cites none of the original papers where the technology was first described.

Finally, several trade and consumer magazines have reviewed firewalls; [Markus 2001] is a web site with links to many of them; the Computer Security Institute (CSI) in San Francisco, California has a search engine for comparing different commercial firewall products [Computer Security Institute 2001].

Several related topics which we do not address thoroughly in our review include intrusion detection systems [Northcutt and Novak 2001; Axelsson 1998], honeypots [Cheswick 1992; Honeynet Project 2001], IPsec implementations [Doraswamy and Harkins 1999], and commercial products.

## 4. THE APPLICATION LEVEL

The application layer contains programs that interact directly with a user. Many application-level protocols exist, including FTP, the Simple Mail Transport Protocol (SMTP), the HyperText Transport Protocol (HTTP), Telnet, etc. The first published paper to describe filtering network traffic for access control, by Jeffrey



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1043

Keysight, EX-1045
Page 1043

Mogul in [1989], described a system which worked at the application level.

### 4.1 The DEC Securing External Access Link

Mogul described a user-level[1] solution to deciding whether or not to pass packets though a router running the UNIX operating system [Mogul 1989]. The packet filter monitored the protocol, source and destination addresses and ports to decide which packets were allowed to continue. Mogul's implementation did not keep track of the state of TCP connections, nor the pseudo-state that can be applied to some UDP requests and responses. However, his implementation could have been expanded to handle these; performance limitations of the computers of the day might have prevented this.

The technology described by Mogul was applied to Digital Equipment Corporation's (DEC) firewall by Paul Vixie [Ranum 1992] and further extended by Marcus Ranum when it became the Securing External Access Link (SEAL)[2], which was the first commercial firewall [Ranum 2001; Avolio 1999], delivered on June 13, 1991 [Avolio 1999]. The DEC SEAL consisted of three components (pictured in Figure 1, part A):

***Gatekeeper.*** The application proxy[3] server for users who were allowed to access external services. **Gatekeeper** was also an externally-accessible machine for uses such as anonymous FTP, the Domain Name System (DNS), etc.

***Gate.*** A packet filtering router limiting what traffic was allowed to pass between the local and external network. This router was configured so that all traffic to/from the inside went to a proxy on **gatekeeper**. *screend*, as described in Mogul's paper [Mogul 1989], performed the packet filtering.

***Mailgate.*** The internal mail server/mail router; this machine was not accessible from the outside. Instead, mail from the outside is delivered to **gatekeeper**, which passes it to **mailgate**.

External connections are permitted only to **gatekeeper**, and it may (depending on the configuration) also be the only way internal users are allowed to access services external to the network.

**Gatekeeper** ran application-level proxies. These proxies also ensured that the proper protocol was used (e.g., that FTP traffic is what went to port 20, and not a Telnet session). The proxies on **gatekeeper** were [Ranum 1992; Treese and Wolman 1993]:

[1] As opposed to a kernel-level.
[2] Some documentation referred to it as the Screening External Access Link.
[3] Proxies were described in detail on page 3.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1044

Keysight, EX-1045
Page 1044

—email and USENET news (since these are store-and-forward protocols, no special handling was needed)

—Telnet

—FTP

—WHOIS (a protocol for looking up network contact information)

—*sup* (a software distribution system developed at Carnegie Mellon University)

—the X window system. X is an especially hard protocol to secure, since the client and server are "backwards": the server is on the desktop and the client is a remote machine. Treese and Wolman at DEC Cambridge succeeded in developing a proxy which needed no changes to the X client or server code at all [Treese and Wolman 1993].

All access across the firewall was through these application proxies. The proxies provided several services [Digital Equipment Corporation 1992]:

—They validated that the source and destination machines were allowed to communicate with each other.

—They ensured that the user was allowed to access the service and remote machine specified.

—They maintained an audit trail—all successful and unsuccessful connections were logged.

—They enforced that the protocol behaved as expected (e.g., that *telnet* was not used to connect to an interactive access terminal server which happened to be running on the FTP port).

When a user wanted to send traffic outside of the local network, the traffic was passed through **gate** to the proxy on **gatekeeper**. If the traffic was allowed, **gatekeeper** then sent the traffic on to the destination machine. No traffic was allowed to pass directly from the local machine to the remote machine (and correspondingly, no traffic was allowed to pass directly from the remote machine to the local machine).

The DEC firewall did not allow any UDP traffic to pass through it at all. This prohibition was probably for two reasons:

—Some of the UDP protocols at the time (e.g., NTP, DNS) could have a server running on **gatekeeper** to which the clients connected.

—Other UDP-based protocols, such as the Network File System (NFS), would have security risks if they were allowed through the firewall.

The DEC SEAL is a classic example of a firewall which uses application-level proxies. Proxies at this level provide excellent security and auditing. Also, because



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1045

**Keysight, EX-1045**
**Page 1045**

they are themselves generating the connection to the remote machine, a proxy has no problems knowing which connections are real and which are being spoofed; this is in contrast to some packet filtering firewalls (Section 6).

One drawback of an application proxy is that it must be designed for the specific protocol. New protocols are developed frequently, requiring the development of new proxies; if there is no proxy, there is no access. Often this time lag creates friction with the users, who may be demanding access to the latest applications, an example of the classic tradeoff between security and user convenience.

Application proxies have other drawbacks. In order to use them, the client program must be changed to accommodate the proxy, telling it about the actual packet destination and authenticating directly to the proxy. Because source code is not publicly available for some applications, in many cases these changes can be made only by the application's vendor, a significant bottleneck. A final drawback of application proxies is performance related, because each packet requires two trips through the complete network protocol stack. None of the papers about the DEC firewall discussed performance directly.

Over time, the DEC SEAL evolved into the AltaVista firewall. The architecture appears to have remained similar, although it was collapsed onto one machine, probably to stay cost-competitive with its competition [Mariet 1997; Smith et al. 1997], resulting in an architecture hybrid of Figure 1, parts B and D. Today, the AltaVista firewall appears to no longer exist.

## 5. THE TRANSPORT LEVEL

In the Internet, the transport level consists of only two protocols, TCP and UDP. This small number of protocols makes writing a proxy much easier—one proxy suffices for all protocols which use TCP. Contrast this with the application-level proxies, where a separate proxy is required for each service, e.g., Telnet, FTP, HTTP, SMTP, etc.

Transport-level proxies retain the advantage that a machine outside of the firewall cannot send packets through the firewall which claim to be a part of an established connection (some of the packet filters described in Section 6 have this problem). Because the state of the TCP connection is known by the firewall, only packets which are a legitimate part of a communication are allowed inside of the firewall.

### 5.1 AT&T's Transport-Level Firewall

While DEC was building SEAL, Dave Presotto wrote and Bill Cheswick re-wrote a firewall for AT&T [Cheswick 1990]. AT&T's firewall also relied on proxies, a term first used in the context of firewalls by Bill Cheswick in [Cheswick 1990]. The resulting firewall was similar in effect to that of Figure 1, part B, although it was



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1046

Keysight, EX-1045
Page 1046

composed of multiple machines.

In contrast to the application-level proxies in the DEC SEAL, the AT&T proxies worked at the transport level [Cheswick 2001]. This difference arose from the different policies at DEC and AT&T. The DEC policy limited outbound connections, with the goal of preventing unauthorized leaks of information to the outside. AT&T did not limit outbound access other than requiring it to pass through the proxy.

As with application-level proxies, users paid a performance penalty because each datagram had to traverse the network protocol stack twice. Cheswick noted that file transfers without the proxy ran at 60-90 Kb/sec, while through the proxy they ranged from 17-44 Kb/sec. Like application-level proxies, transport-level proxies require that the client program be modified to use the proxy (however, modifying a program to use a transport-level proxy is easier than modifying it to use an application-level proxy; see SOCKS in Section 5.2).

### 5.2 Later Proxies

Kolbas and Kolbas wrote SOCKS [Koblas and Koblas 1992; Leech et al. 1996; Leech 1996; McMahon 1996] with the goal of simplifying the use of proxies. A SOCKS call replaced a normal socket call, which resulted in all outbound traffic using the proxy. This approach was a nice, clean solution, and worked well if one had the source code for the relevant operating system utilities. Some commercial applications (e.g., Netscape) also were written to accommodate SOCKS. A system using SOCKS and TCP connections could be transparent to the user (assuming the proxy allowed them access to their destination host). In [2000], Fung and Chang describe an enhancement to SOCKS to allow it to handle UDP streams, such as that used by RealNetworks' RealPlayer.

Marcus Ranum and Frederick Avolio worked on the Trusted Information Systems (TIS) Firewall Toolkit (FWTK), a collection of proxies for building firewalls [Ranum and Avolio 1994; Avolio and Ranum 1994]. This freely-available toolkit provided SMTP, the Network News Transport Protocol (NNTP), FTP and Telnet application proxies as well as a generic circuit-level proxy. To improve security, the proxies used the UNIX system call *chroot* to limit how much of the system became visible; this way if a proxy were compromised, the rest of the firewall was more likely to remain trustworthy. The TIS FWTK had no proxies for UDP services; instead, the firewall machine ran DNS and the Network Time Protocol (NTP). The inside machines used the firewall for those services. When Trusted Information Systems and Network Associates, Inc. (NAI) merged in February 1998, the TIS firewall became NAI's Gauntlet Internet Firewall.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1047

Keysight, EX-1045
Page 1047

## 6. THE NETWORK LEVEL

If a limited number of protocols at the transport level is an advantage, then it would seem that the single protocol, IP, at the network level would be even better. In some ways, it is. Packet filtering is fast, and it does not require the knowledge or cooperation of the users. However, a packet filter working only at the network level means that it cannot determine whether a packet belongs to an established communication or not. And, similarly to transport-level proxies, it cannot enforce which application-level protocol is used. For organizations trying to prevent an information leak, this is a serious limitation. It is relatively straightforward to use allowed protocols to send arbitrary data; for an example, see [daemon9 and Alhambra 1996] which describes communication using the Internet Control Message Protocol (ICMP) echo ("ping") and ICMP echo reply ("pong") packets. In spite of these impediments, packet filtering at the IP level has been successful when state information about the connection is used (Section 6.2).

### 6.1 Packet Filtering

Packet filtering for network security began with Mogul's paper about *screend* [1989]. Whereas application- and transport-level proxies require each datagram to make a trip up and down the whole protocol stack, packet filtering can be much faster. Most of the early papers on packet filtering for security stressed the performance benefits [Corbridge et al. 1991; Chapman 1992; Bailey et al. 1994; Molitor 1995]; later papers continued this trend [Suri and Varghese 1999; Lyu and Lau 2000]. Besides the performance benefits, packet filtering does not require the cooperation of the users, nor does it require any special action on their part [Weber 1999].

Packet filters use one or more of the following pieces of information to make their decision on whether or not to forward the packet [Reed 2002a]:

—source address

—destination address

—options in the network header

—transport-level protocol (i.e., TCP, UDP, ICMP, etc.)

—flags in the transport header

—options in the transport header

—source port or equivalent if the protocol has such a construct

—destination port or equivalent if the protocol has such a construct

—the interface on which the packet was received or will be sent

—whether the packet is inbound or outbound



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1048

Keysight, EX-1045
Page 1048

Although packet filtering is fast, it does have drawbacks. The most important limitation is the difficulty of writing correct filters. Several researchers have pointed out this problem and attempted to solve it. Mogul in [1991] discusses the difficulty of setting up packet filtering rules and provides correct examples as illustrations. In [1992], Chapman makes the point that most packet filter languages are similar to assembly language. In [1995], Molitor describes an improved commercial filter language.

Another drawback of packet filtering is that it cannot determine which user is causing which network traffic. It can inspect the IP address of the host where the traffic originates, but a host is not the same as a user. If an organization with a packet-filtering firewall is trying to limit the services some users can access, it must either implement an additional, separate protocol for authentication or use the IP address of the user's primary machine as a weak replacement for true user authentication.

Also, because IP addresses can be spoofed, using them for authentication can lead to other problems. If the router is running a properly configured filter, remote attackers should not be able to spoof local addresses, but they could spoof other remote addresses. Local machines can spoof other local machines easily. In spite of these problems, many organizations are using IP addresses or DNS names for access control.

When a proxy is used, the connection to the remote machine comes from the machine running the proxy. With packet filters, the local machine directly initiates the connection to the remote machine. A result of this difference is that the entire internal network is potentially reachable from external connections; otherwise the reply packets from the remote host would not be delivered properly. As a consequence, hostile remote computers may be able to exploit weaknesses in the protocol implementation of the local machine (e.g., [Securityfocus.com 1997]).

Protocols such as FTP [Postel and Reynolds 1985] also cause problems for packet filters. FTP uses a control channel opened from the client to the server for commands. However, when getting a file, one method of using FTP (active FTP) has the server open a connection back to the client. This connection is initiated from the FTP data port (TCP port 20) to a random client port. Since malicious programs could use the FTP data port as their source port, packet filter writers cannot assume that a connection from TCP port 20 is a FTP transfer. One solution is to use passive FTP, where the client opens the connection for the data transfer to the server. However, not all FTP clients support passive transfers. RFC 1579 [Bellovin 1994] suggests that FTP client writers include support for the passive method of file transfer, but this solution has yet to be universally adopted.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1049

Keysight, EX-1045
Page 1049

### 6.2 Packet Filtering with State

Originally, packet filters did not keep track of the state of a connection. This means that a remote host could send in packets which appeared to be part of an established TCP connection (with the TCP ACK flag set), but which in reality were not. This allowed a remote attacker to map the local network as if the firewall did not even exist. Attacks against bugs in the TCP/IP protocol stack (e.g., [Securityfocus.com 1997]) can pass through the firewall by claiming to be part of an established TCP session.

The solution to this problem is to keep track of the state of a connection, a property referred to variously as stateful firewalls, adaptive firewalling and packet inspection. In other words, the packet filter looks at both the network level and the transport level data. The router monitors the initial TCP packets with the SYN flag set and allows the packets to return through only until the FIN packet is sent and acknowledged. A similar pseudo-state can be kept for most UDP (e.g., DNS, NTP) and some ICMP communication (e.g., ping)—a request sent out opens a hole for the reply, but only for a short time. In [1992], Chapman was one of the first to point out the problem of the lack of state in packet filtering firewalls. In [1995], Molitor identified the problem of implementing state in his packet filtering architecture as a "future direction." The first peer-reviewed paper to describe adding state to packet filters was by Julkunen and Chow in [1998], where they describe a dynamic packet filter for Linux. This paper was proceeded by a technical report in [1997].

Although Julkunen and Chow were the first to publish a peer-reviewed paper about keeping state in packet-filtering firewalls, they were not the first to keep track of connection state. IP Filter (IPF) version 3.0 in 1996 by Darren Reed predated their work [Reed 2002b; 2002c]. The first peer-reviewed description of this work appeared much later in a discussion of state in IPF [van Rooij 2001].

### 6.3 Improving Packet Filter Specification

In [1992], Chapman makes the point that writing packet filter rules is similar to writing in assembly language. Some researchers have therefore developed higher-level languages for specifying packet filters. These improved packet filter languages are simpler than the policy specifications mentioned in Section 10, and they do somewhat ease the writing of filter rules. Specific examples include:

—In [2000], Hazelhurst proposed binary decision diagrams (BDDs) for visualizing router rule sets. Since BDDs represent boolean expressions, they are ideal for representing the block/pass rules which occur in packet filters. BDDs also make automated analysis of packet filter rules easier, as well as providing better performance than the table lookups used in many routers.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1050

Keysight, EX-1045
Page 1050

—The filter language compiler, *flc* [Reed 19], allows the use of the C preprocessor, specification of a default block or pass policy for various directions of traffic flow, and provides a simple if-then-else facility. *flc* also generates rules for several different packet filters (IPF, *ipfw*, *ipfwadm*, *ipfirewall*, Cisco extended access lists, and *screend*).

### 6.4 Network Address Translation (NAT)

Because the Internet is short of IPv4 addresses, many people use Network Address Translation (NAT) to gain more mileage out of a single IP address [Egevang and Francis 1994; Srisuresh and Egevang 2001; Hain 2000]. When a router uses NAT, it changes the source address of outbound packets to its own address (or one from a pool of addresses which it controls). It chooses a local, unused port for the upper layer protocol (TCP or UDP), and stores in a table the association between the new address and port and the real sender's address and port. When the reply arrives, it looks up the real destination in this table, rewrites the packet, and passes it to the client. When the connection is finished (or after a timeout period for UDP packets), the entry is removed from the table.

NAT provides a similar form of protection to that of proxies. In NAT, all connections originate from the router performing the address translation. As a result, someone outside the local network cannot gain access to the protected local machines unless the proper entry exists in the table on the router. The network administrator can manually install such an entry, causing all traffic destined for a specific port to be forwarded to a server for that service (in effect, providing an Internet-accessible service on an inside machine.[4]).

RFC 2663 [Srisuresh and Holdrege 1999] notes that NAT is not without its problems. For example, NAT may prevent IPsec from working correctly. One feature of IPsec is the ability to know that a packet was not modified during transit. However, one of the purposes of NAT is to modify the packet—the source address and possibly the source port must be modified for NAT to work. DNS problems can also occur. A machine behind a router using NAT has a name and an IP address. However, most networks using NAT also use RFC1918 [Rekhter et al. 1996] private IP addresses, which are not globally unique. Therefore, DNS inside the network is not meaningful outside the network.

[4]Setting up such an entry is usually a bad idea from a security standpoint. Maintaining a server inside a firewall is risky because if it is compromised, the attacker then has access inside the network, which as noted in Section 2 is likely to be insecure.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1051

Keysight, EX-1045
Page 1051

## 7. THE DATA-LINK LAYER

A bridge is a network device which works at the ISO data-link layer. Because it operates at this level, it does not need to access to routing information. A bridging firewall uses the information listed in Section 6.1 to decide whether or not to block a packet. As a result, a bridging firewall can look at data in several other levels of the Internet Protocol suite, including the network and transport layers. Because a filtering bridge is still a filter, the disadvantages of packet filtering still apply to it.

What makes a bridging firewall different from a packet filtering router is that it can be placed anywhere because it is transparent at the network level (see Figure 1, part D). It can be used to protect a single machine or a small collection of machines that would normally not warrant a separate network required when using a router. As it does not need its own IP address, the bridge itself can be immune to any attack which makes use of IP (or any of the protocols on top of IP). And, no configuration changes are needed in the protected hosts when the firewall is installed. Installation times can be minimal (for example, Limoncelli claims three second install times [Limoncelli 1999]), so users are minimally disrupted when the bridge is installed.

The first bridging firewalls were described by Kahn et al. in [1997] and developed for computers running DOS. They refer to earlier unpublished research concerning packet filtering filtering on a bridge. Later work which also explores the idea of a bridging firewall includes Jianbing Liu and Yan Ma in [Liu and Ma 1999]. Keromytis and Wright discuss using IPsec on a bridging firewall to set up secure, virtual LANs [2000].

## 8. OTHER APPROACHES TO FIREWALLS

Some firewall designs do not fit cleanly into a single network protocol layer. Some of them (e.g. Normalization, discussed in Section 8.2) are ideas that can be implemented in a firewall using one of the technologies already discussed. These proposals are just now being incorporated in firewall implementations. Others, such as the distributed firewalls discussed in Section 8.5 are more revolutionary, changing the entire architecture of the firewall. Although a few organizations may be using firewalls based on these more revolutionary ideas, they are not yet in widespread use.

### 8.1 Transparent Proxies

As mentioned in Sections 4.1 and 5.1, the primary problem with proxies has been that the client software must be modified and/or the user must work differently when using the. Transparent proxies address this limitation. With a transparent proxy the client sends packets to the destination in a normal fashion. However, when the packets reach the firewall, access control checks and logging are performed as in



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1052

Keysight, EX-1045
Page 1052

a classical proxy system. The "magic" is implemented by the firewall, which notes the destination address and port, opens up a connection to it and then replies to the client, as if the proxy were the remote machine. This relaying can take place at either the transport level or the application level. RFC 1919 [Chatel 1996] compares classical proxies with transparent proxies. Bonachea and McPeak use transparent proxies to improve the security of FTP in [2001]. Rodriguez et al. describe what they call translucent proxying of TCP in [2001].

Transparent proxies are demanding because the firewall must operate both at the network and application levels. Therefore, performance has remained a concern. One solution proposed by Spatscheck et al. [2000] and Maltz and Bhagwat [1998] is that of "splicing." In splicing, after the proxy verifies that communication is allowed to proceed, the firewall converts to a network-level packet filtering firewall for that communication. Splicing provides the extra control of proxies but maintains performance closer to that of packet filters.

### 8.2 Normalization

Attackers often abuse protocol specifications, e.g., by sending overlapping IP fragments or out-of-order TCP byte sequences. In [2001], Handley et al. stressed that a firewall is a good location for enforcing tight interpretation of a protocol. Besides protecting the computers behind the firewall from attacks based on protocol abuses, this so-called "normalization" also makes signature-based intrusion detection systems more reliable because they see a consistent data stream. Handley et al. provide a list of possible normalizations, ranging from those guaranteed to be safe to others that are potentially too strict in their interpretation of the standard. They were not the first to suggest normalization, however. In [2000], Malan et al. describe "transport scrubbing," and more recently the idea is elaborated in [Watson et al. 2001]. At about the same time, Strother [Strother 2000] proposed a similar idea. Her solution involved different rings of trust, in which a network packet must pass through one ring before proceeding to the next. Many of her rings achieve the same effect as normalization.

### 8.3 Signature-based Firewalls

Malan et al. discuss "application scrubbing" in [2000]. In this approach, a user-level program is established as a transparent proxy (see Section 8.1) which monitors the data stream for strings known to be hazardous (and presumably to prevent these strings from reaching the client). Watson et al. refer to the same concept as a "fingerprint scrubber" in [2001].

*Snort* [Roesch 1999] is a common intrusion detection system. *Hogwash* [Larsen 2001] is a firewall that blocks packets matching the *snort* rules. It runs on a bridging



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1053

Keysight, EX-1045
Page 1053

firewall (Section 7) and the authors claim it can handle network speeds of up to 100Mbps on hardware which is not state-of-the-art.

### 8.4 Firewalls at Other ISO Network Levels

The Common Object Request Broker (CORBA) allows applications written in one language to make requests of objects which may be written in a different language and which may be available on a different machine. The CORBAgate by Dotti and Rees in [1999a; 1999b] is a presentation-level proxy. When a request is made to an object which is on the other side of the firewall, the proxy transparently changes the references. The result is that objects on either side of the firewall end up referring to an object on the firewall.

### 8.5 Distributed Firewalls

There are several limitations to the firewall technology that we have presented so far. One common assumptions is that all the hosts inside a firewall are trustworthy. This assumption is not always valid—for example, see the discussion about the problems with virtual private networks (VPNs) in Section 14.1. A related problem is that firewalls ignore internal traffic, which may violate security policy. Because firewalls are typically centralized in one location, they can become performance bottlenecks as well as providing a single point of failure. A further limitation of conventional firewalls arises because in some cases the local machines know the context of the communication not available to the firewall. For example, a file transfer may be allowed or denied based on what file is being transferred and by whom. The firewall does not have this local, contextual knowledge.

One solution to these problems, proposed by Bellovin [1999], is a distributed firewall. This was implemented by Ioannidis et al. in [2000] and by Markham and Payne in [2001]. In this firewall, the network administrator has a single policy specification, loaded onto all machines. Each host runs its own local firewall implementing the policy. Machines are identified by cryptographic certificates, a stronger form of authentication than IP addresses. With a distributed firewall, the common concept of a DMZ or screened network, where servers accessible to the outside world are located, is no longer necessary (for an example of a DMZ or screened network, see Figure 1, part C).

Hwang and Gangadharan [Hwang and Gangadharan 2001; Gangadharan and Hwang 2001] propose using firewalls on all devices attached to the network, where they can be combined with an intrusion detection system (IDS). When the IDS detects an anomalous event, it modifies the firewall to react to the threat. Lower overhead can be achieved with this approach than that reported for the distributed firewall developed by Ioannidis [2000]. However, the architecture still uses a con-



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1054

Keysight, EX-1045
Page 1054

ventional firewall.

Distributed firewalls have a different set of problems than centralized ones. The most significant is that a distributed firewall relies on its users (with physical access to the machine) not to override or replace the policy. In other words, the network administrator must trust his or her users. Additionally, if the firewall is running as a part of the operating system, then the operating system must protect the firewall software. However, the local firewall is protecting the operating system, creating a circular set of dependencies. In [2001], Markham and Payne propose implementing the distributed firewall on a programmable network interface card (NIC) to reduce reliance on the operating system for protection.

Around the same time that Bellovin proposed the distributed firewall, Ganger and Nagle at Carnegie Mellon University also proposed a distributed approach to security in [2000], in which each device was responsible for its part of the security policy. Ganger and Nagle argue that if each device were more secure, then an attacker who succeeds in passing the outer defenses (the firewall) would not find vulnerable targets inside. They propose installing security devices on many parts of a network, such as NICs, storage devices, display devices, and networking hardware such as routers and switches. The idea is that the devices would dynamically adjust their approach to security based on the overall network defense level. As with Bellovin's proposal, programmable NICs are an important part of the overall strategy.

### 8.6 Protection against denial of service attacks

In a denial of service attack, the attacker's goal is to reduce or eliminate an authorized user's access to a service, machine or network by disabling the service or machine, or by overloading some aspect of the system with voluminous but legal activities. Because all traffic for the network travels through the firewall, some denial-of-service attacks can be stopped at the firewall. For example, protocol-based attacks (e.g., [Computer Emergency Response Team (CERT) 1997b] and [Computer Emergency Response Team (CERT) 1998]), which cause the remote machine to crash, can be stopped by protocol normalization (Section 8.2). When attackers are attempting to saturate available bandwidth, Ioannidis and Bellovin proposed the idea of pushback—the ability for a router to detect and preferentially drop packets which are part of an attack, and also to notify upstream routers of the attack [Ioannidis and Bellovin 2002].

In [2000], Balasubramanian described an architecture in which network- and host-based intrusion detection systems are combined with a policy-driven coordinator to respond to denial of service attacks. This response is a job that traditionally would be given to the firewall. As presented, the response could be anything a process



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1055

Keysight, EX-1045
Page 1055

can do; it is up to the network administrator to determine the proper response to the attacks. Balasubramanian recognizes but does not address the problem of false positives or spoofed network packets being used to cause a reaction desired by the attacker.

In [1997], Schuba et al. review various approaches to SYN-flood denial-of-service attacks and present a new approach (not limited only to firewalls). They suggest using a program to monitor all traffic on the LAN. They describe a program which categorizes SYN packets into four classes: never seen, belonging to correctly behaving hosts, potentially spoofed, and certainly spoofed. In addition to the program's classification, the administrator can provide a collection of addresses known to be good or bad. When the program sees SYN packets from bad hosts (dynamically characterized by their behavior), it sends a RST packet to kill the half-open connection.

### 8.7 Multicast

On the Internet, multicast is often used for various forms of multimedia. In contrast with traditional unicast communication, in multicast the sender does not necessarily know the identities of all the participants in the session. And, this is also true for the recipients, who do not know in advance all the possible people who might be sending to them. This difference also makes proxies such as SOCKS difficult to implement unless they change the multicast into a collection of unicasts, a change that defeats the benefits of multicast—with multicast, once one client inside of the firewall has joined a group, others may join without needing to authenticate. Additionally, the multicast routing protocol, the Internet Group Management Protocol (IGMP), specifies only multicast groups and not UDP ports; in a default configuration, a multicast source has access to the complete set of UDP ports on client machines. If a source has access to all UDP ports, then it could potentially attack other services, (e.g. Microsoft networking) which are unrelated to the service it is providing.

The classic paper on multicast and firewalls was published by Djahandari and Sterne in [1997]. In this paper they describe an application proxy for the TIS Firewall Toolkit. The proxy has the following features:

—It allows authentication and auditing;

—It prevents multicast traffic from reaching hosts which did not request it;

—It allows the multicast traffic to be sent only to "safe" ports.

The proxy converts multicast traffic into unicast traffic. Unfortunately, this approach also means that it does not scale well, as a collection of $N$ users all receiving the same multicast stream increases the traffic inside the firewall by a factor of $N$ over what it would have been if multicast had been retained. On the other hand,



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1056

Keysight, EX-1045
Page 1056

they do solve all of the security problems mentioned in the previous paragraph and later in this section.

RFC 2588 [Finlayson 1999] suggests several possible solutions to the problem of multicast and firewalls. For example, communication between external and internal machines could be tunneled through the firewall using the UDP Multicast Tunneling Protocol (UMTP). This protocol was designed to connect clients to the Multicast Backbone (the MBone), but would work for tunneling through multicast-unaware firewalls.

RFC 2588 also mentions the possibility of dynamic firewall rules, and in [1999], Oria describes in further detail how they can be implemented. A program runs on the router, which monitors multicast session announcements. The program reads the announcements, and if the specified group and UDP port are allowed by the policy, it generates the necessary rules permitting the data to pass through the firewall. When a client informs the router that it wishes to join a multicast group, it sends an IGMP join message to the router. The dynamically generated rules permit or deny this access. This approach to multicast on the firewall assumes that session announcements can be trusted. Unfortunately, this is not a valid assumption because they can be spoofed.

### 8.8 Transient Addressing

Many protocols, such as FTP, RealAudio, and H.323 (a protocol used for programs such as Microsoft's NetMeeting), open secondary channels for additional communication. These additional channels are a problem for firewalls unless the firewall makes use of a stateful proxy. In [2001] Gleitz and Bellovin propose a solution to this problem by taking advantage of version 6 of the Internet Protocol (IPv6), which has 128 bits of address space. This is large enough for each host to have multiple addresses. A client initiating a connection to a FTP server uses an address which includes the process group ID of the FTP client process. The firewall sees a connection from a specific IPv6 address going to a FTP server at a remote site, and then allows all communication from the server back to the client's address. On the client side, this address is only used for the FTP process; connections from the FTP server to other ports on the client will not be accepted, because only the FTP client is using that specific address.

## 9. COMMERCIAL FIREWALL PRODUCTS

It is difficult to separate completely advances in firewall technology from the commercial products that implement them. There is a large market for commercial firewall products, which has driven many important recent developments. At the same time, without direct inspection of the source code, it can be quite difficult



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1057

Keysight, EX-1045
Page 1057

to ascertain exactly how or why a particular product operates. Further, the success or failure of commercial products depends as much on business issues as on technical merit. In spite of these problems, no review of firewall technology would be complete without some discussion of commercial products. In the following, we highlight a few of the many products currently on the market, emphasizing those that we believe are interesting intellectually, have had unusual impact on the development of firewalls, or are particularly clear examples of a particular methodology.

### 9.1 Network Associates, Inc. (NAI)

As mentioned in Section 5.2, the TIS firewall toolkit was an early firewall based on proxies. After the merger with NAI, it became the basis for the Gauntlet firewall. This firewall continued to evolve and now incorporates the following features [Network Associates Technology, Inc. 2001b; 2001c; 2001d]:

—Packet filtering with state;

—Transport-level proxies;

—Transparent application-level proxies for common protocols such as RealAudio, QuickTime, HTTP, SMTP, and FTP;

—Groups to ease the packet filter specification;

—Splicing like was described in Section 8.1 to improve performance;

—Content scanning for known viruses and hostile applications.

### 9.2 Cisco

Cisco has two firewall products (IOS firewall and Private Internet Exchange (PIX)) which share many features, but also have some differences. Both are based on packet filtering with state. They also perform what is known as "inspection," in which the packet filter inspects the data inside the application layer and modifies the packet filtering rules to adapt to what the application is expected to do. For example, the inspection algorithm could determine that a FTP client will need a port opened, changing the filtering rules appropriately. The result of such packet inspection is similar to that provided by a transparent application proxy, but with less overhead. It also suffers from the same problems—as protocols evolve, the inspection code must also evolve in order to take advantage of the new features in the protocol.

Both firewalls can require authentication against Terminal Access Controller Access Control System (TACACS)+ or Remote Authentication Dial-In User Service (RADIUS) before passing packets from or to specified hosts and/or services. The IOS firewall can also use Kerberos for authentication [Cisco Systems, Inc. 2001].



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1058

Keysight, EX-1045
Page 1058

The PIX is especially notable because it was the first commercially available implementation of NAT [Cisco Systems, Inc. 2000]. It also regenerates TCP sequence numbers, using strong random numbers. This is important because machines which have weak TCP sequence number generation algorithms are vulnerable to session hijacking and spoofed one-way connections.

The IOS firewall protects against pre-identified malicious Java applets, presumably by matching against a static signature file. The IOS firewall defends and protects against SYN floods and other closely related TCP attacks. As explained in [Cisco Systems, Inc. 2001], SYN-floods are detected by

> comparing the rate of requests for new connections and the number of half-open connections to a configurable threshold level to detect SYN flooding. When the router detects unusually high rates of new connections, it issues an alert message and takes a preconfigured action

The preconfigured action either instructs the protected host to drop half-open connections (presumably by sending it a TCP reset packet), or temporarily filters all TCP SYN packets destined for the protected host. Unfortunately, this action disables all inbound connections to the protected host. The IOS firewall also monitors TCP sequence numbers and drops all packets with suspicious numbers.

### 9.3 Lucent's firewall family

Lucent's firewall family [Lucent Technologies 2001] is based on packet filtering with state. Similar to Cisco, packets are inspected, and the packet filter examines at the higher-level data (including some protocols at the application layer). Unlike the Cisco products, the firewall can be a bridging firewall.

### 9.4 Sun's Sunscreen Secure Net

Sun's Sunscreen Secure Net 3.1 [Sun Microsystems 2000] is based on packet filtering with state and a collection of application proxies. Its HTTP proxy can be configured to pass or drop Java applets. It can also filter:

—Java applets based on the cryptographic signature included in the applet,

—Java applets based on a hash of the applet,

—Cookie requests, and

—ActiveX content.

The FTP proxy is a true application proxy and therefore can limit access to certain file transfer commands such as `put` or `get` based on source and destination addresses and/or user authentication.

Sunscreen Secure Net can also be configured as a bridging firewall.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1059

**Keysight, EX-1045**
**Page 1059**

## 10. FIREWALL POLICY SPECIFICATION

Firewalls were originally built and configured by experts. Now, firewalls are commodity products which are sold with the intent that nearly anyone can be responsible for their network's security. The network administrator uses a graphical user interface (GUI) to configure packet filtering rules. Unfortunately, this GUI still requires the user to understand the complexities of packet filters. These complexities were originally pointed out in [1992] by Chapman. In many cases, the only difference between then and now is that the user interacts with the packet filter rules through a GUI. The common use of transparent proxies only increases the complexity of the administrator's task.

Some researchers have tried to improve the way that firewalls are managed. Guttman [1997] described a LISP-like language for expressing access control policies for networks where more than one firewall router is used to enforce the policy. The language is then used to compute a set of packet filters which will properly implement the policy. He also presents an algorithm for comparing previously generated filters to the policy to identify any policy breaches. However, the automatically generated filters are not expressed in the language of any router; the network administrator must build them manually from the LISP-like output.

In [2000], Hazelhurst et al. describe binary decision diagrams (BDDs) for visualizing router rule sets. BDDs have the benefit that they can represent boolean expressions—this makes them ideal for representing the block/pass rules which occur in packet filters. Once a set of packet filter rules has been converted to BDDs, it becomes easy to apply automated analysis. BDDs are also an efficient method for storing and using the rules; they can be used by a router to provide better performance than the simple table-based lookup which is used in many routers in 2002.

The Internet standards RFC2748 [Durham et al. 2000], RFC3060 [Moore et al. 2001] and RFC3084 [Chan et al. 2001] describe the Common Open Policy Service (COPS) protocol. This protocol is used between a policy server (Policy Decision Point or PDP) and its clients (Policy Enforcement Points or PEPs). The basic idea is that the policy is specified at a different location from the firewall (a PEP), and the policy server ensures that the various policy enforcers have and use the correct policy. The policy may relate to Quality of Service (QoS), it may relate to security, or it may relate to some other part of network policy (e.g., IPsec); the COPS protocol is extensible. The network is modeled as a finite state machine and a policy is modeled as a collection of policy rules. These rules have a logical expression of conditions and a set of actions. If the logical expression is true, then the actions are executed. These actions may cause a state change in the network finite state machine. The policy rules can be prioritized, allowing conflict resolution



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1060

Keysight, EX-1045
Page 1060

when two or more rules match but specify conflicting actions. As these proposed standards are adopted, they will have a significant impact on how firewalls are constructed.

Stone et al. survey policy languages through late 2000 and describe a new approach to policy specification [Stone et al. 2001]. In addition to security concerns, their approach also takes into account Quality of Service (QoS). In specifying policies, they note that some policies are static, i.e., for security reasons, all access to certain network addresses are prohibited. Other policies, however, are dynamic, i.e., if the available bandwidth is too low, streaming video is no longer allowed. Finally, different users may receive different levels of service (e.g., customers in the company web store have priority over employees browsing the web). Their policy language is called the Path-Based Policy Language (PPL), and it corrects some of the deficiencies in the other policy languages.

Damianou et al. describe a policy language, Ponder, in [2001]. Ponder is a declarative, object-oriented language, which through its structures can represent policies. Constraints on a policy can also be represented in Ponder. Although Ponder appears to be a particularly rich and expressive language for expressing policies, there is not yet an automated policy implementation path.

Bartal et al. describe *firmato* [Bartal et al. 1999]. *firmato* has an underlying entity-relationship model which specifies the global security policy, a language in which to represent the model, a compiler which translates the model into firewall rules, and a tool which displays a graphical view of the result to help the user visualize the model.

A module for use with *firmato* is the firewall analysis engine, Fang (Firewall ANalysis enGine) by Mayer et al. [2000]. Fang reads the firewall configurations and discovers what policy is described. The network administrator can then verify that the policy being implemented by the various routers matches the desired policy. For example, the network administrator can ask questions such as "From which machines can our DMZ be reached, and with which services?" Fang builds a representation of the policy; it is not an active testing program. This difference allows Fang to test both that authorized packets actually succeed as well as that unauthorized packets are blocked. It also allows testing before the firewall is deployed; by contrast, active test tools require the firewall to be up and running to test it. Also, active testing cannot test the network's vulnerability to spoofing attacks, whereas Fang can. Fang provides a GUI to collect the queries from the network administrator and to display the results.

A recent example of this family of firewall test and analysis tools is the Lumeta Firewall Analyzer (LFA) [Wool 2001]. LFA is a commercial product which goes beyond Fang by automatically generating the "interesting" queries. The network



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1061

Keysight, EX-1045
Page 1061

administrator needs only to provide the firewall configuration. The result is a system that hides the complexities of the underlying router configurations, providing a much more comprehensible picture of the resulting policy.

## 11. FIREWALL TESTING

Firewall testing was originally an ad-hoc exercise, the thoroughness being determined by the skill of the person running the tests. A second phase of testing methodology included security scanners such as the Security Administrator Tool for Analyzing Networks (SATAN) [Venema 1995] and the Internet Security Systems (ISS) Internet scanner [Internet Security Systems, Inc. 2002]. These scanners provided the basis for the National Computer Security Association (NCSA) certification [Vigna 1997] for a period of time. Vigna extended this approach by defining a formal model of a network's topology in [1997]. His model can also represent the TCP/IP protocol stack up through the transport level. Using this model, he was able to generate logical statements describing the requirements for the firewall. Given these requirements, he then generated a series of locations for probes and packets to attempt to send when testing the real firewall. From a formal standpoint, this work is promising, but it fails to address the common problem of how to develop a correct formal description. Producing complete formal descriptions for realistic networks represents a significant amount of work and is difficult to do correctly. Additionally, the test generator must have a complete list of vulnerabilities for which to generate tests.

Marcus Ranum took a different approach to firewall testing in [Ranum 1995]; he notes that firewalls are (or at least should be) different for different organizations. After a firewall is deployed, an expert can study the policy specification for the firewall and decide which tests will verify that the firewall properly implements the policy, using a top-down approach. He emphasizes the importance of testing both the security of the firewall itself (that the firewall is secure from attack) and the correctness of the policy implementation. Unfortunately, such testing is both expensive and time-consuming.

Some of the tools for firewall policy specification (Section 10) also provide testing or guidance for testing.

## 12. FIREWALL THEORY

Firewalls have evolved in an ad-hoc fashion, reacting to various attacks. Lyles and Schuba [1996b; 1996a] and Schuba et al. [1997] developed the first formal model for firewalls. Their model described firewall technology in terms of a policy $P$, communication traffic $T$, and a network policy domain $D$. This model allows firewall functions to be distributed across multiple machines. It also allows for



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1062

Keysight, EX-1045
Page 1062

repeated application of the theory; for example, at the link, network, and transport layers of a network. Schuba expanded the model in his Ph.D. dissertation [1997], developing a theoretical basis for firewalls based on Hierarchical Colored Petri Nets (HCPN).

A different and less detailed formal model of firewalls is presented by Vigna in [Vigna 1996; 1997]. Here, a network is modeled as a hypergraph, protocols such as IP, UDP, and TCP are modeled as tuples (possibly containing other tuples). A vulnerability is a boolean expression of quantified boolean expressions relating the tuples and hypergraphs. These vulnerabilities can be used to generate tests for firewalls.

Packet classification is an important part of filtering traffic. Gupta and McKeown wrote a review of packet classification algorithms in [2001]. They provide examples of four different types of algorithms, and discuss when the algorithms might be most useful.

## 13. WHAT FIREWALLS DO NOT PROTECT AGAINST

No firewall provides perfect security. Several problems exist which are not addressed by the current generation of firewalls. In the event that a firewall does try to provide protection for the problems discussed in this section, either it is not in widespread use or it has problems with the protection it provides.

### 13.1 Data Which Passes Through the Firewall

A firewall is probably best thought of as a permeable membrane. That is, it is only useful if it allows some traffic to pass through it (if not, then the network could be physically isolated from the outside world and the firewall not needed). Unfortunately, any traffic passing though the firewall is a potential avenue of attack. For example, most firewalls have some provision for email, but email is a common method of attack; a few of the many email attacks are described in [Cohen et al. 2001; Computer Emergency Response Team (CERT) 1999; 2000a; 2001b; 2001c; 2001d]. The serious problem of email-based attacks has resulted in demand for some part of the firewall to check email for hostile code. Products such as AMaViS [amavis.org 2002] and commercial email virus scanners (e.g., [Network Associates Technology, Inc. 2001a]) have responded to this problem. However, they are only as good as the signatures for which they scan; novel attacks pass through without a problem.

If web traffic is allowed through the firewall, then network administrators must cope with possibility of malicious web sites. With scripting languages such as Java, JavaScript, and ActiveX controls, malicious web administrators can read arbitrary files on client machines (e.g., [Hernan 2000]) and execute arbitrary code on the client



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1063

Keysight, EX-1045
Page 1063

(e.g., [Cohen and Hernan 2000; Computer Emergency Response Team (CERT) 1997a]). ActiveX controls are of particular concern, because they do not run in any form of "sandbox" the way Java applets do [Bellovin et al. 2000]. ActiveX controls can be digitally signed, and if properly used, can be used to authenticate the author, if not the author's intentions.

In [1997], Martin et al. describe some attacks written in Java. They advocate the draconian solution of blocking all applets, on the grounds that it cannot be determined which Java applets are dangerous. They suggest the following methods of blocking Java applets at the firewall:

(1) Using a proxy to rewrite `<applet>` tags. This requires that the proxy be smart enough to rewrite only the tags in HTML files and not if they appear in other file types, such as image files. This requires that the proxy parse the HTML documents in the same manner as the browser.

(2) Java class files always begin with four byte hex signature CAFE BABE. A firewall could block all files that begin with this sequence. A possibility of false positives exists with this scheme, but Martin et al. believe that this problem is less likely to occur than the `<applet>` tag appearing in non-HTML files.

(3) Block all files whose names end in `.class`. This solution is weak because Java classes can come in files with other extensions, for example, packing class files in a `.zip` file is common.

Their suggestion is to implement all three of these, and they write a proxy which does everything except look inside of `.zip` files.

### 13.2 Servers on the DMZ

Because the networks inside of a firewall are often not secure, servers which must be accessible from the Internet (e.g., web and mail servers) are often placed on a screened network, called the DMZ (for demilitarized zone; for a picture of one way a DMZ may be constructed, see Figure 1, part C). Machines on the DMZ are not allowed to make connections to machines on the inside of the firewall, but machines on the inside *are* allowed to make connections to the DMZ machines. The reason for this architecture is that if a server on the DMZ is compromised, the attacker cannot directly attack the other machines inside. Because a server must be accessible to be of use, current firewalls other than signature-based ones (Section 8.3) can do little against attacks through the services offered. Examples of attacks on servers include worms such as those described in [Danyliw and Householder 2001; Danyliw et al. 2001].



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1064

Keysight, EX-1045
Page 1064

### 13.3 Insider Attacks

In spite of the fact that early firewalls such as the DEC SEAL were initially set up to prevent information leaks, they cannot protect against insiders intent on getting information out of an organization. Consider a hostile employee with access to a CD burner. The resulting CD will not be traveling through the firewall, so the firewall cannot prevent this data loss. In [1995], Muffett also points out that inside a firewall, security tends to decrease over time unless the internal machines are continually updated. Therefore, a hostile insider can generally penetrate other internal machines, and since these attacks do not go through the firewall, it cannot stop them. To reduce this threat, some organizations have set up internal firewalls.

## 14. FUTURE CHALLENGES FOR FIREWALLS

All of the topics discussed in the prior section pose serious challenges for firewalls. In addition, two emerging technologies will further complicate the job of a firewall, Virtual Private Networks (VPNs) and peer-to-peer networking.

### 14.1 VPNs

Because firewalls are deployed at the network perimeter, if the network perimeter is expanded the firewall must somehow protect this expanded territory. VPNs provide an example of how this can happen. A laptop being used by a traveling employee in an Internet cafe or a home machine which is connected to an ISP via a DSL line or cable modem must be inside the firewall. However, if the laptop or home machine's security is breached, the entire internal network becomes available to the attackers.

Remote access problems are first mentioned in [Avolio and Ranum 1994]. Due to the fact that VPNs had not yet been invented, it is easy to understand why Avolio and Ranum failed to discuss the problem of a remote perimeter which includes hosts always-connected to the Internet (via DSL or cable modems) and which are also allowed inside through a VPN tunnel.

### 14.2 Peer-to-peer Networking

The music sharing system Napster is the most famous example of peer-to-peer networking. However, several other peer-to-peer systems exist as well, including Gnutella (e.g., [Wego Systems, Inc 2001]) and AIMster (file sharing over AOL Instant Messenger). When not used for music sharing, peer-to-peer file sharing is used to support collaboration between distant colleagues. However, as Bellovin points out in [2000], these systems raise serious security concerns. These include the possibility of using Gnutella for attacks, buggy servents (server+client programs), and the problems of web and email-based content in yet another form. Current



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1065

Keysight, EX-1045
Page 1065

firewalls are unable to provide any protection against these types of attacks beyond simply blocking the peer-to-peer networking.

## 15. CONCLUSION

The need for firewalls has led to their ubiquity. Nearly every organization connected to the Internet has installed some sort of firewall. The result of this is that most organizations have some level of protection against threats from the outside. Attackers still probe for vulnerabilities that are likely to only apply to machines inside of the firewall. They also target servers, especially web servers. However, these attackers are also now targeting home users (especially those with full-time Internet connections) who are less likely to be well protected.

Because machines inside a firewall are often vulnerable to both attackers who breach the firewall as well as hostile insiders, we will likely see increased use of the distributed firewall architecture. The beginnings of a simple form of distributed firewalls are already here, with personal firewalls being installed on individual machines. However, many organizations will require that these individual firewalls respond to configuration directives from a central policy server. This architecture will simply serve as the next level in a sort of arms race, as the central server and the protocol(s) it uses become special targets for attackers.

Firewalls and the restrictions they commonly impose have affected how application-level protocols have evolved. Because traffic initiated by an internal machine is often not as tightly controlled, newer protocols typically begin with the client contacting the server; not the reverse as active FTP did. The restrictions imposed by firewalls have also affected the attacks that are developed. The rise of email-based attacks is one example of this change.

An even more interesting development is the expansion of HTTP and port 80 for new services. File sharing and remote procedure calls can now be accomplished using HTTP. This overloading of HTTP results in new security concerns, and as a result, more organizations are beginning to use a (possibly transparent) web proxy so they can control the remote services used by the protected machines. The future is likely to see more of this co-evolution between protocol developers and firewall designers until the protocol designers consider security when the protocol is first developed. Even then, firewalls will still be needed to cope with bugs in the implementations of these protocols.

## REFERENCES

ABIE, H. 2000. An overview of firewall technologies. *Telektronikk 96,* 3, 47–52. `http://www.nr.no/publications/FirewallTechnologies.pdf` Accessed 2002 Feb 20.

AMAVIS.ORG. 2002. AMaViS—a mail virus scanner. `http://www.amavis.org/` Accessed 2002 Feb 20.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1066

**Keysight, EX-1045**
**Page 1066**

Avolio, F. 1999. Firewalls and Internet security, the second hundred (Internet) years. *The Internet Protocol Journal 2,* 2 (June), 24–32. http://www.cisco.com/warp/public/759/ipj_2-2/ipj_2-2_fis1.html Accessed 2002 Feb 20.

Avolio, F. M. and Ranum, M. J. 1994. A network perimeter with secure external access. In *Internet Society Symposium on Network and Distributed Systems Security, 3-4 Feb. 1994, San Diego, CA, USA*. Internet Society, Reston, VA, USA, 109–119. http://www.ja.net/CERT/Avolio_and_Ranum/isoc94.ps Accessed 2002 Feb 20.

Axelsson, S. 1998. Research in intrusion-detection systems: A survey. Tech. Rep. 98–17, Dept. of Computer Eng. Chalmers Univ. of Tech, SE-412 96 Göteborg, Sweden. December. http://www.ce.chalmers.se/staff/sax/survey.ps Accessed 2002 Feb 20.

Bailey, M. L., Gopal, B., Pagels, M. A., Peterson, L. L., and Sarkar, P. 1994. PATHFINDER: A pattern-based packet classifier. In *1st Symposium on Operating Systems Design and Implementation, 14-17 November 1994, Monterey, CA, USA*. USENIX Association, Berkeley, CA, 115–123.

Balasubramanian, S. 2000. An architecture for protection of network hosts from denial of service attacks. M.S. thesis, University of Florida. http://etd.fcla.edu/etd/uf/2000/ana6124/Master.pdf Accessed 2002 Feb 20.

Bartal, Y., Mayer, A. J., Nissim, K., and Wool, A. 1999. Firmato: A novel firewall management toolkit. In *1999 IEEE Symposium on Security and Privacy, 9-12 May 1999, Oakland, CA, USA*. IEEE, Los Alamitos, CA, USA, 17–31. http://www.wisdom.weizmann.ac.il/~kobbi/papers/firmato.ps Accessed 2002 Feb 20.

Bellovin, S. 1994. Firewall-friendly FTP. RFC 1579. ftp://ftp.isi.edu/in-notes/rfc1579.txt Accessed 2002 Feb 20.

Bellovin, S., Cohen, C., Havrilla, J., Herman, S., King, B., Lanza, J., Pesante, L., Pethia, R., McAllister, S., Henault, G., Goodden, R., Peterson, A. P., Finnegan, S., Katano, K., Smith, R., and Lowenthal, R. 2000. Results of the security in ActiveX workshop Pittsburgh, Pennsylvania USA August 22-23, 2000. Tech. rep., CERT Coordination Center, Software Engineering Institute, Carnegie Mellon University, Pittsburg, PA 15213, USA. December. http://www.cert.org/reports/activeX_report.pdf Accessed 2002 Feb 20.

Bellovin, S. M. 1992. There be dragons. In *UNIX Security Symposium III Proceedings, 14-16 Sept 1992, Baltimore, MD, USA*. USENIX Association, Berkeley, CA, 1–16. http://www.research.att.com/~smb/papers/dragon.pdf Accessed 2002 Feb 20.

Bellovin, S. M. 1999. Distributed firewalls. *;login: 24,* Security (November). http://www.usenix.org/publications/login/1999-11/features/firewalls.htm%l Accessed 2002 Feb 20.

Bellovin, S. M. 2000. Security aspects of napster and gnutella. Invited talk at the USENIX 2001 Annual Technical Conference, June 25-30, 2001. http://www.research.att.com/~smb/talks/NapsterGnutella/index.htm Accessed 2002 Feb 20.

Bonachea, D. and McPeak, S. 2001. SafeTP: Transparently securing FTP network services. Tech. Rep. UCB Tech Report CSD-01-1152, Computer Science Division, University of California, Berkeley, 387 Soda Hall #1776, Berkeley, CA 94720-1776. http://www.cs.berkeley.edu/~bonachea/safetp/CSD-01-1152.pdf Accessed 2002 Feb 20.

Braden, R., Clark, D., Crocker, S., and Huitema, C. 1994. Report of IAB workshop on



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1067

Keysight, EX-1045
Page 1067

security in the Internet architecture February 8-10, 1994. RFC 1636. ftp://ftp.isi.edu/in-notes/rfc1636.txt Accessed 2002 Feb 20.

Chan, K., Seligson, J., Durham, D., Gai, S., McCloghrie, K., Herzog, S., Reichmeyer, F., Yavatkar, R., and Smith, A. 2001. COPS usage for policy provisioning (COPS-PR). RFC 3084. ftp://ftp.isi.edu/in-notes/rfc3084.txt Accessed 2002 Feb 20.

Chapman, D. B. 1992. Network (in)security through IP packet filtering. In *UNIX Security Symposium III Proceedings, 14-16 Sept 1992, Baltimore, MD, USA*. USENIX Association, Berkeley, CA, 63–76. http://www.greatcircle.com/pkt_filtering.html Accessed 2002 Feb 20.

Chatel, M. 1996. Classical versus transparent IP proxies. RFC 1919. ftp://ftp.isi.edu/in-notes/rfc1919.txt Accessed 2002 Feb 20.

Cheswick, B. 1990. The design of a secure Internet gateway. In *USENIX 1990 Summer Conference*. USENIX Association, Berkeley, CA. http://www.cheswick.com/ches/papers/gateway.ps Accessed 2002 Feb 20.

Cheswick, B. 1992. An evening with Berferd in which a cracker is lured, endured, and studied. In *Winter 1992 USENIX Conference, 20-24 Jan 1992, San Francisco, CA, USA*. USENIX Association, Berkeley, CA, 163–173. http://www.cheswick.com/ches/papers/berferd.ps Accessed 2002 Feb 20.

Cheswick, B. 2001. Personal communication.

Cheswick, W. R. and Bellovin, S. M. 1994. *Firewalls and Internet Security: Repelling the Wily Hacker*. Addison-Wesley, One Jacob Way, Reading, MA, 01867.

Cisco Systems, Inc. 2000. Cisco - Cisco's PIX firewall and stateful firewall security. http://www.cisco.com/warp/public/cc/pd/fw/sqfw500/tech/nat_wp.htm Accessed 2002 Feb 20.

Cisco Systems, Inc. 2001. Cisco - building a perimeter security solution. http://www.cisco.com/warp/public/cc/pd/iosw/ioft/iofwft/tech/firew_wp.h%tm Accessed 2002 Feb 20.

Cohen, C., Danyliw, R., Finlay, I., Shaffer, J., Hernan, S., Houle, K., King, B. B., and van Ittersum, S. 2001. CERT advisory CA-2001-03 VBS/OnTheFly (Anna Kournikova) malicious code. http://www.cert.org/advisories/CA-2001-03.html Accessed 2002 Feb 20.

Cohen, C. and Hernan, S. 2000. CERT advisory CA-2000-12 HHCtrl ActiveX control allows local files to be executed. http://www.cert.org/advisories/CA-2000-12.html Accessed 2002 Feb 20.

Computer Emergency Response Team (CERT). 1997a. CERT advisory CA-1996-07 weaknesses in Java bytecode verifier. http://www.cert.org/advisories/CA-1996-07.html Accessed 2002 Feb 20.

Computer Emergency Response Team (CERT). 1997b. CERT advisory CA-1996-26 denial-of-service attack via ping. http://www.cert.org/advisories/CA-1996-26.html Accessed 2002 Feb 20.

Computer Emergency Response Team (CERT). 1998. CERT advisory CA-1997-28 IP denial-of-service attacks. http://www.cert.org/advisories/CA-1997-28.html Accessed 2002 Feb 20.

Computer Emergency Response Team (CERT). 1999. CERT advisory CA-1999-04 Melissa macro virus. http://www.cert.org/advisories/CA-1999-04.html Accessed 2002 Feb 20.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1068

Keysight, EX-1045
Page 1068

Computer Emergency Response Team (CERT). 2000a. CERT advisory CA-2000-04 love letter worm. http://www.cert.org/advisories/CA-2000-04.html Accessed 2002 Feb 20.

Computer Emergency Response Team (CERT). 2000b. CERT incident note IN-2000-02 : Exploitation of unprotected windows networking shares. http://www.cert.org/incident_notes/IN-2000-02.html Accessed 2002 Feb 20.

Computer Emergency Response Team (CERT). 2000c. CERT incident note IN-2000-03 : 911 worm. http://www.cert.org/incident_notes/IN-2000-03.html Accessed 2002 Feb 20.

Computer Emergency Response Team (CERT). 2001a. CERT incident note IN-2001-01 : Widespread compromises via "ramen" toolkit. http://www.cert.org/incident_notes/IN-2001-01.html Accessed 2002 Feb 20.

Computer Emergency Response Team (CERT). 2001b. Vulnerability note VU#131569: Microsoft Outlook view control allows execution of arbitrary code and manipulation of user data. http://www.kb.cert.org/vuls/id/131569 Accessed 2002 Feb 20.

Computer Emergency Response Team (CERT). 2001c. Vulnerability note VU#149424 Outlook Web Access (OWA) executes scripts contained in email attachment opened via Microsoft Internet Explorer (IE). http://www.kb.cert.org/vuls/id/149424 Accessed 2002 Feb 20.

Computer Emergency Response Team (CERT). 2001d. Vulnerability note VU#5648: Buffer overflows in various email clients. http://www.kb.cert.org/vuls/id/5648 Accessed 2002 Feb 20.

Computer Security Institute. 2001. CSI firewall product search center. http://www.spirit.com/CSI/firewalls.html Accessed 2002 Feb 20.

Corbridge, B., Henig, R., and Slater, C. 1991. Packet filtering in an IP router. In *Proceedings of the USENIX Fifth System Administration Conference (LISA V)*. USENIX Association, Berkeley, CA, 227–232. http://www.netsys.com/library/papers/packet_filtering_in_an_ip_router.p%df Accessed 2002 Feb 20.

daemon9 and Alhambra. 1996. Project Loki: ICMP tunneling. *Phrack 7,* 49 (August). http://www.phrack.org/show.php?p=49&a=6 Accessed 2002 Feb 20.

Damianou, N., Dulay, N., Lapu, E., and Sloman, M. 2001. The ponder policy specification language. In *Policies for Distributed Systems and Networks. International Workshop, POLICY 2001. Proceedings, 29-31 Jan. 2001, Bristol, UK*. Springer-Verlag, Berlin, Germany. http://www.doc.ic.ac.uk/~mss/Papers/Ponder-Policy01V5.pdf Accessed 2002 Feb 20.

Danyliw, R., Dougherty, C., Householder, A., and Ruefle, R. 2001. CERT advisory CA-2001-26 Nimda worm. http://www.cert.org/advisories/CA-2001-26.html Accessed 2002 Feb 20.

Danyliw, R. and Householder, A. 2001. CERT advisory CA-2001-19 "code red" worm exploiting buffer overflow in IIS indexing service DLL. http://www.cert.org/advisories/CA-2001-19.html Accessed 2002 Feb 20.

Denning, P. J. March-April 1989. The Internet worm. *American Scientist 77,* 2, 126–128.

Digital Equipment Corporation. 1992. Securing external access link (SEAL) introductory guide. This document was part of the documentation provided by DEC to purchasers of SEAL.

Djahandari, K. and Sterne, D. 1997. An MBone proxy for an application gateway firewall. In



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1069

Keysight, EX-1045
Page 1069

*Proceedings of the 1997 Conference on Security and Privacy (S&P-97)*. IEEE Press, Los Alamitos, 72–81.

Doraswamy, N. and Harkins, D. 1999. *IPSec: the new security standard for the Internet, intranets, and virtual private networks*. Prentice-Hall, Upper Saddle River, NJ.

Dotti, P. and Rees, O. 1999a. Protecting the hosted application server. In *Proceedings of the IEEE 8th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*. IEEE Computer Society, Los Alamitos, CA, USA.

Dotti, P. and Rees, O. 1999b. Protecting the hosted application server. Tech. Rep. HPL-1999-54 990413, Hewlett-Packard Labs., Bristol, UK. `http://www.hpl.hp.com/techreports/1999/HPL-1999-54.pdf` Accessed 2002 Feb 20.

Durham, D., Boyle, J., Cohen, R., Herzog, S., Rajan, R., and Sastry, A. 2000. The COPS (Common Open Policy Service) protocol. RFC 2748. `ftp://ftp.isi.edu/in-notes/rfc2748.txt` Accessed 2002 Feb 20.

Egevang, K. and Francis, P. 1994. The IP network address translator (NAT). RFC 1631. `ftp://ftp.isi.edu/in-notes/rfc1631.txt` Accessed 2002 Feb 20.

Eichin, M. W. and Rochlis, J. A. 1989. With microscope and tweezers: An analysis of the Internet virus of November 1988. In *1989 IEEE Computer Society Symposium on Security and Privacy*. IEEE Computer Society, Los Alamitos, CA, USA, 326–343.

Farrow, R. 1998. CSI 1997 firewalls matrix [an analysis of current firewall technologies]. `http://www.spirit.com/CSI/Papers/farrowpa.htm` Accessed 2002 Feb 20.

Finlayson, R. 1999. IP multicast and firewalls. RFC 2588. `ftp://ftp.isi.edu/in-notes/rfc2588.txt` Accessed 2002 Feb 20.

Fung, K. P. and Chang, R. K. C. 2000. A transport-level proxy for secure multimedia streams. *IEEE Internet Computing 4,* 6 (November–December), 57–67.

Gangadharan, M. and Hwang, K. 2001. Intranet security with micro-firewalls and mobile agents for proactive intrusion response. In *Proceedings of the International Conference on Computer Networks and Mobile Computing, 2001*. IEEE Computer Society, Los Alamitos, CA, USA, 325–332.

Ganger, G. R. and Nagle, D. F. 2000. Enabling dynamic security management of networked systems via device-embedded security. Tech. Rep. CMU-CS-00-174, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213-3890. December. `http://reports-archive.adm.cs.cmu.edu/anon/2000/CMU-CS-00-174.pdf` Accessed 2002 Feb 20.

Gleitz, P. M. and Bellovin, S. M. 2001. Transient addressing for related processes: Improved firewalling by using IPV6 and multiple addresses per host. In *Proceedings of the 10th USENIX Security Symposium*. USENIX Association, Berkleley, CA, USA, 99–113. `http://www.usenix.org/publications/library/proceedings/sec01/full_paper%s/gleitz/gleitz.pdf` Accessed 2002 Feb 20.

Gupta, P. and McKeown, N. 2001. Algorithms for packet classification. *IEEE Network 15,* 2 (March-April), 24–32.

Guttman, J. D. 1997. Filtering postures: local enforcement for global policies. In *1997 IEEE Symposium on Security and Privacy, 4-7 May 1997, Oakland, CA, USA*. IEEE Computer Society Press, Los Alamitos, CA, USA, 120–129. `http://www.mitre.org/support/papers/filtering_postures/filtering_postures.pdf` Accessed 2002 Feb 20.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1070

**Keysight, EX-1045**
**Page 1070**

Hain, T. 2000. Architectural implications of NAT. RFC 2993. ftp://ftp.isi.edu/in-notes/rfc2993.txt Accessed 2002 Feb 20.

Hambridge, S. and Sedayao, J. C. 1993. Horses and barn doors: Evolution of corporate guidelines for Internet usage. In *Proceedings of the USENIX Seventh System Administration Conference (LISA '93)*. USENIX Association, Berkeley, CA. http://www.usenix.org/publications/library/proceedings/lisa93/full_pape%rs/hambridge.ps Accessed 2002 Feb 20.

Handley, M., Paxson, V., and Kreibich, C. 2001. Network intrusion detection: Evasion, traffic normalization, and end-to-end protocol semantics. In *Conference Proceedings: 10th USENIX Security Symposium*. USENIX Association, Berkleley, CA, USA, 115–131. http://www.aciri.org/vern/papers/norm-usenix-sec-01.pdf Accessed 2002 Feb 20.

Hazelhurst, S., Attar, A., and Sinnappan, R. 2000. Algorithms for improving the dependability of firewall and filter rule lists. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN 2000)*. IEEE Computer Society, Los Alamitos, CA, USA. IEEE.

Hernan, S. 2000. CERT advisory CA-2000-15 Netscape allows Java applets to read protected resources. http://www.cert.org/advisories/CA-2000-15.html Accessed 2002 Feb 20.

Honeynet Project. 2001. *Know Your Enemy: Revealing the Security Tools, Tactics, and Motives of the Blackhat Community*. Addison-Wesley, Boston, MA.

Hwang, K. and Gangadharan, M. 2001. Micro-firewalls for dynamic network security with distributed intrusion detection. In *IEEE International Symposium on Network Computing and Applications, NCA 2001*. IEEE Computer Society, Los Alamitos, CA, USA, 68–79.

Internet Security Systems, Inc. 2002. ISS - security products - security assessment - Internet scanner. http://www.iss.net/securing_e-business/security_products/security_asses%sment/internet_scanner/ Accessed 2002 Feb 20.

Ioannidis, J. and Bellovin, S. M. 2002. Pushback: Router-based defense against DDoS attacks. In *Proceedings of Network and Distributed System Security Symposium, Catamaran Resort Hotel San Diego, California 6-8 February 2002*. The Internet Society, 1775 Wiehle Ave., Suite 102, Reston, VA 20190. http://www.research.att.com/~smb/papers/pushback-impl.pdf Accessed 2002 Feb 20.

Ioannidis, S., Keromytis, A. D., Bellovin, S. M., and Smith, J. M. 2000. Implementing a distributed firewall. In *ACM Conference on Computer and Communications Security*. Association for Computing Machinery, One Astor Plaza, 1515 Broadway, New York, New York 10036-5701, USA, 190–199. http://www.cis.upenn.edu/~angelos/Papers/df.ps.gz Accessed 2002 Feb 20.

Julkunen, H. and Chow, C. 1997. Enhance network security with dynamic packet filter. Tech. Rep. EAS-CS-97-2, University of Colorado at Colorado Springs Department of Computer Science, 1420 Austin Bluffs Parkway, P.O. Box 7150, Colorado Springs, CO 80933-7150. April. http://www.cs.auc.dk/~fleury/papers/firewall/julkunen.pdf Accessed 2002 Feb 20.

Julkunen, H. and Chow, C. 1998. Enhance network security with dynamic packet filter. In *7th International Conference on Computer Communications and Networks, 12-15 Oct. 1998, Lafayette, LA, USA*, K. Makki, I. Chalamrac, and N. Pissinou, Eds. IEEE, Piscataway, NJ, USA, 268–275.

Kahn, A., Al-Darwish, N., Guizani, M., Menten, M., and Youssef, H. 1997. Design and implementation of a software bridge with packet filtering and statistics collection functions.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1071

Keysight, EX-1045
Page 1071

*International Journal of Network Management 7,* 5 (September-October), 251–263. http://www1.acm.org/pubs/articles/journals/ijnm/1997-7-5/p251-khan/p251%-khan.pdf Accessed 2002 Feb 20.

Keromytis, A. D. and Wright, J. L. 2000. Transparent network security policy enforcement. In *FREENIX Track. 2000 USENIX Annual Technical Conference, 18-23 June 2000, San Diego, CA, USA*. USENIX Association, Berkeley, CA, USA, 215–225. http://www.cis.upenn.edu/~angelos/Papers/bridgepaper.ps.gz Accessed 2002 Feb 20.

Koblas, D. and Koblas, M. R. 1992. SOCKS. In *UNIX Security Symposium III Proceedings, 14-16 Sept. 1992, Baltimore, MD, USA*. USENIX Association, Berkeley, CA, 77–83.

Larsen, J. 2001. HogWash. http://hogwash.sourceforge.net/ Accessed 2002 Feb 20.

Leech, M. 1996. Username/password authentication for SOCKS V5. RFC 1929. ftp://ftp.isi.edu/in-notes/rfc1929.txt Accessed 2002 Feb 20.

Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., and Jones, L. 1996. SOCKS protocol version 5. RFC 1928. ftp://ftp.isi.edu/in-notes/rfc1928.txt Accessed 2002 Feb 20.

Lightoler, T. 1764. *The gentleman and farmer's architect. A new work. Containing a great variety of ... designs. Being correct plans and elevations of parsonage and farm houses, lodges for parks, pinery, peach, hot and green houses, with the fire-wall, tan-pit, &c particularly described ...* R. Sayer, London, UK.

Limoncelli, T. 1999. Tricks you can do if your firewall is a bridge. In *1st Conference on Network Administration, 7-10 April 1999, Santa Clara, CA, USA*. USENIX Association, Berkeley, CA, USA, 47–55. http://www.bell-labs.com/user/tal/papers/ Accessed 2002 Feb 20.

Liu, J. and Ma, Y. 1999. Packet filtering in bridge. In *1999 Internet Workshop (WS'99), 18-20 Feb. 1999, Osaka, Japan*. IEEE, Piscataway, NJ, USA, 94–98.

Lodin, S. W. and Schuba, C. L. 1998. Firewalls fend off invasions from the net. *IEEE Spectrum 35,* 2 (February), 26–34.

Lucent Technologies. 2001. Lucent - VPN firewall family. http://www.lucent.com/products/solution/0,,CTID+2012-STID+10080-SOID+12%01-LOCL+1,00.html Accessed 2002 Feb 20.

Lyles, J. and Schuba, C. 1996a. A reference model for firewall technology and its implications for connection signaling. Tech. Rep. CSD-TR-96-073, Department of Computer Sciences, Purdue University. December.

Lyles, J. B. and Schuba, C. L. 1996b. A reference model for firewall technology and its implications for connection signaling. Tech. Rep. CSD-TR-94-061, Reprinted as Department of Computer Sciences, Purdue University, Purdue University, 1398 Computer Science Building, West Lafayette, IN 47907-1398. Proceedings Open Signaling Workshop, Columbia University, New York, NY, October 1996. https://www.cerias.purdue.edu/techreports-ssl/public/csd_94-061.pdf Accessed 2002 Feb 20.

Lyu, M. R. and Lau, L. K. 2000. Firewall security: Policies, testing and performance evaluation. In *Proceedings of The Twenty-Fourth Annual International Computer Software and Applications Conference*. IEEE Computer Society, Los Alamitos, CA, USA.

Malan, G. R., Watson, D., Jahanian, F., and Howell, P. 2000. Transport and application protocol scrubbing. In *IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies,*



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1072

**Keysight, EX-1045**
**Page 1072**

*26-30 March 2000, Tel Aviv, Israel*. IEEE Computer Society; IEEE Communications Society, Piscataway, NJ, USA, 1381–1390.

MALTZ, D. AND BHAGWAT, P. 1998. TCP splicing for application layer proxy performance. Tech. Rep. RC 21139, IBM. March. http://domino.watson.ibm.com/library/cyberdig.nsf/a3807c5b4823c53f85256%561006324be/88d1e552b09ffa65852565e6006616f1?0penDocument Accessed 2002 Feb 20.

MARIET, P. 1997. AltaVista firewall. In *DECUS Switzerland conference*. Decus Switzerland, Sonnenbergstrasse 11 CH-8610 Uster. http://elias.decus.ch/presentations/ge_19970415_av/index.htm Accessed 2002 Feb 20.

MARKHAM, T. AND PAYNE, C. 2001. Security at the network edge: a distributed firewall architecture. In *DARPA Information Survivability Conference & Exposition II, 2001. DISCEX '01. Proceedings*. Vol. 1. IEEE Computer Society, Los Alamitos, CA, USA, 279–286.

MARKUS, H. S. 2001. Firewall guide software reviews. http://www.firewallguide.com/software.htm Accessed 2002 Feb 20.

MARTIN, D.M., J., RAJAGOPALAN, S., AND RUBIN, A. 1997. Blocking Java applets at the firewall. In *SNDSS '97: Internet Society 1997 Symposium on Network and Distributed System Security, 10-11 Feb. 1997, San Diego, CA, USA*. IEEE Computer Society Press, Los Alamitos, CA, USA.

MAYER, A., WOOL, A., AND ZISKIND, E. 2000. Fang: A firewall analysis engine. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy (S&P 2000)*. IEEE Computer Society, Los Alamitos, CA, USA.

MCKAY, N. 1998. China: The great firewall. Web publication: http://www.wired.com/news/politics/0,1283,16545,00.html Accessed 2002 Feb 20.

MCMAHON, P. 1996. GSS-API authentication method for SOCKS version 5. RFC 1961. ftp://ftp.isi.edu/in-notes/rfc1961.txt Accessed 2002 Feb 20.

MOGUL, J. 1991. Using *screend* to implement IP/TCP security policies. Tech. Rep. NSL Technical Note TN-2, Digital Network Systems Laboratory, Palo Alto, CA. July. http://research.compaq.com/nsl/publications/postscript/nsltn2.pdf Accessed 2002 Feb 20.

MOGUL, J. C. 1989. Simple and flexible datagram access controls for Unix-based gateways. In *Proceedings of the USENIX Summer 1989 Conference*. USENIX Association, Berkeley, CA, 203–222. ftp://ftp.digital.com/pub/Digital/WRL/research-reports/WRL-TR-89.4.ps.g%z Accessed 2002 Feb 20.

MOLITOR, A. 1995. An architecture for advanced packet filtering. In *Proceedings of the Fifth USENIX UNIX Security Symposium*. USENIX Association, Berkeley, CA, USA, 117–126. http://www.usenix.org/publications/library/proceedings/security95/full_%papers/molitor.ps Accessed 2002 Feb 20.

MOORE, B., ELLESSON, E., STRASSNER, J., AND WESTERINEN, A. 2001. Policy core information model—version 1 specification. RFC 3060. ftp://ftp.isi.edu/in-notes/rfc3060.txt Accessed 2002 Feb 20.

MUFFETT, A. 1994. Proper care and feeding of firewalls. In *Proceedings of the UKERNA Computer Security Workshop*. United Kingdom Education and Research Networking Association, Atlas Centre, Chilton, Didcot, Oxfordshire, OX11 0QS UK. ftp://coast.cs.purdue.edu/pub/doc/firewalls/Muffett_Alec_feeding_firewa%lls.ps.Z Accessed 2002 Feb 20.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1073

**Keysight, EX-1045**
**Page 1073**

Muffett, A. 1995. Wan-hacking with *AutoHack* - auditing security *behind* the firewall. In *The Fifth USENIX UNIX Security Symposium*. USENIX Association, Berkeley, CA, 21–34.

Network Associates Technology, Inc. 2001a. E-mail protection. `http://www.mcafeeb2b.com/products/email-protection.asp` Accessed 2002 Feb 20.

Network Associates Technology, Inc. 2001b. Gauntlet firewall administrator's guide version 6.0. `http://download.nai.com/products/media/pgp/support/gnt/admin_GNT60.pdf` Accessed 2002 Feb 20.

Network Associates Technology, Inc. 2001c. Gauntlet firewall for UNIX advanced administrator's guide version 6.0. `http://download.nai.com/products/media/pgp/support/gnt/adv_admin_60.pdf` Accessed 2002 Feb 20.

Network Associates Technology, Inc. 2001d. Gauntlet firewall services guide version 6.0. `http://download.nai.com/products/media/pgp/support/gnt/services_60.pdf` Accessed 2002 Feb 20.

Northcutt, S. and Novak, J. 2001. *Network Intrusion Detection: An Analyst's Handbook, Second Edition*. New Riders Publishing, 201 West 103rd St, Indianapolis, IN, 46290, USA.

Oria, L. 1999. Approaches to multicast over firewalls: an analysis. Tech. Rep. HPL-IRI-1999-004 990827, Hewlett-Packard Laboratories. August. `http://www.hpl.hp.com/techreports/1999/HPL-IRI-1999-004.html` Accessed 2002 Feb 20.

Postel, J. and Reynolds, J. 1985. File transfer protocol (FTP). RFC 959. `ftp://ftp.isi.edu/in-notes/rfc959.txt` Accessed 2002 Feb 20.

Ranum, M. J. 1992. A network firewall. In *Proceedings of the First World Conference on System Administration and Security*. SANS Institute, 5401 Westbard Ave. Suite 1501, Bethesda, MD 20816.

Ranum, M. J. 1995. On the topic of firewall testing. `http://web.ranum.com/pubs/fwtest/index.htm` Accessed 2002 Feb 20.

Ranum, M. J. 2001. personal communication.

Ranum, M. J. and Avolio, F. M. 1994. A toolkit and methods for Internet firewalls. In *Conference Proceedings: USENIX Summer 1994 Technical Conference*. USENIX Association, Berkeley, CA, 37–44.

Reed, D. 19?? Filter language compiler. Undated web page. `http://coombs.anu.edu.au/ipfilter/flc.html` Accessed 2002 Feb 20.

Reed, D. 2002a. IP filter. Undated web page. `http://coombs.anu.edu.au/~avalon/ip-filter.html` Accessed 2002 May 15.

Reed, D. 2002b. IP filter. HISTORY file from the IP Filter distribution. `http://coombs.anu.edu.au/~avalon/ip-fil3.4.27.tar.gz` Accessed 2002 May 15.

Reed, D. 2002c. What's new for IP filter. Undated web page. `http://coombs.anu.edu.au/~avalon/ipfil-new.html` Accessed 2002 May 15.

Rekhter, Y., Moskowitz, B., Karrenberg, D., deGroot, G. J., and Lear, E. 1996. Address allocation for private internets. RFC 1918. `ftp://ftp.isi.edu/in-notes/rfc1918.txt` Accessed 2002 Feb 20.

Rodriguez, P., Sibal, S., and Spatscheck, O. 2001. TPOT: translucent proxying of TCP. *Computer Communications 24,* 2, 249–255. `http://www.terena.nl/conf/wcw/Proceedings/S7/S7-3.pdf` Accessed 2002 Feb 20.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1074

**Keysight, EX-1045**
**Page 1074**

ROESCH, M. 1999. Snort - lightweight intrusion detection for networks. In *LISA'99: 13th System Administration Conference and Exhibition, 7-12 Nov. 1999, Seattle, WA, USA*. USENIX Association, Berkleley, CA, USA, 229–238. `http://www.usenix.org/events/lisa99/full_papers/roesch/roesch.pdf` Accessed 2002 Feb 20.

SCHIMMEL, J. 1997. A historical look at firewall technologies. *;login: 22,* 7. `http://www.usenix.org/sage/best.of/breakins/firewall.html` Accessed 2002 Feb 20.

SCHNEIER, B. 2000. *Secrets and Lies: Digital Security in a Networked World*. John Wiley & Sons, New York, NY, 188–193.

SCHUBA, C., LYLES, B., AND SPAFFORD, E. 1997. A reference model for firewall technology. In *13th Annual Computer Security Applications Conference (ACSAC), 8-12 Dec. 1997, San Diego, CA, USA*. IEEE Computer Society, Los Alamitos, CA, USA, 133–145.

SCHUBA, C. L. 1997. On the modeling, design, and implementation of firewall technology. Ph.D. thesis, Purdue University. `ftp://ftp.cerias.purdue.edu/pub/papers/christoph_schuba/schuba_phddis.p%df` Accessed 2002 Feb 20.

SCHUBA, C. L., KRSUL, I. V., KUHN, M. G., SPAFFORD, E. H., SUNDARAM, A., AND ZAMBONI, D. 1997. Analysis of a denial of service attack on TCP. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*. IEEE Computer Society, IEEE Computer Society Press, Los Alamitos, CA, USA, 208–223. `https://www.cerias.purdue.edu/techreports-ssl/public/97-06.ps` Accessed 2002 Feb 20.

SECURITYFOCUS.COM. 1997. Multiple vendor "out of band" data (winnuke.c) DoS vulnerability. Vulnerability database. `http://www.securityfocus.com/bid/2010` Accessed 2002 Feb 20.

SMITH, J. M., DOHERTY, S. G., LEAHY, O. J., AND TYNAN, D. M. 1997. Protecting a private network: The AltaVista firewall. *Digital Technical Journal 9,* 2 (November), 17–32. `http://www.research.compaq.com/wrl/DECarchives/DTJ/DTJQ00/index.html` Accessed 2002 Feb 20.

SPAFFORD, E. H. 1988. The Internet worm program: An analysis. Tech. Rep. Purdue Technical Report CSD-TR-823, Department of Computer Science, Purdue University, West Lafayette, IN 47907-2004. `ftp://ftp.cs.purdue.edu/pub/reports/TR823.PS.Z` Accessed 2002 Feb 20.

SPAFFORD, E. H. 1991. The Internet worm incident. Tech. Rep. Purdue Technical Report CSD-TR-933, Department of Computer Science, Purdue University, West Lafayette, IN 47907-2004.

SPATSCHECK, O., HANSEN, J. S., HARTMAN, J. H., AND PETERSON, L. L. 2000. Optimizing TCP forwarder performance. *IEEE/ACM Transactions on Networking 8,* 2, 146–157. `http://www.cs.arizona.edu/scout/Papers/TR98-01.ps` Accessed 2002 Feb 20.

SRISURESH, P. AND EGEVANG, K. 2001. Traditional IP network address translator (traditional NAT). RFC 3022. `ftp://ftp.isi.edu/in-notes/rfc3022.txt` Accessed 2002 Feb 20.

SRISURESH, P. AND HOLDREGE, M. 1999. IP network address translator (NAT) terminology and considerations. RFC 2663. `ftp://ftp.isi.edu/in-notes/rfc2663.txt` Accessed 2002 Feb 20.

STOLL, C. 1989. *The Cuckoo's Egg*. Doubleday, New York, NY.

STOLL, C. May, 1988. Stalking the wily hacker. *Commun. ACM 31,* 5, 484–497.

STONE, G. B., LUNDY, B., AND XIE, G. G. 2001. Network policy languages: A survey and a new approach. *IEEE Network 15,* 1 (January-February), 10–21.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1075

**Keysight, EX-1045**
**Page 1075**

Strother, E. 2000. Denial of service protection: the nozzle. In *Annual Computer Security Applications Conference, 11-15 Dec. 2000, New Orleans, LA, USA*. IEEE Computer Society, Los Alamitos, CA, USA, 32–41. http://www.acsac.org/2000/papers/41.pdf Accessed 2002 Feb 20.

Sun Microsystems, I. 2000. Sunscreen 3.1 reference manual. http://www.sun.com/software/securenet/ds/ Accessed 2002 Feb 20.

Suri, S. and Varghese, G. 1999. Packet filtering in high speed networks. In *Tenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'99)*. SIAM, 3600 University City Science Center, Philadelphia, PA 19104-2688, 969–970. http://siesta.cs.wustl.edu/~suri/psdir/soda_filter.ps Accessed 2002 Feb 20.

The RFC Editor. 2001. Request for comments (RFC) frequently asked questions. http://www.rfc-editor.org/rfcfaq.html Accessed 2002 Feb 20.

Treese, G. W. and Wolman, A. 1993. X through the firewall and other application relays. In *Proceedings of the USENIX Summer Conference*. USENIX Association, Berkeley, CA. ftp://crl.dec.com/pub/DEC/CRL/tech-reports/93.10.ps.Z Accessed 2002 Feb 20.

van Rooij, G. 2001. Real stateful TCP packet filtering in IP-filter. Invited talk at the 10th USENIX Security Symposium August 13-17, 2001 Washington DC. Talk slides available at: http://www.usenix.org/events/sec01/invitedtalks/rooij.pdf.

Venema, W. 1995. Project SATAN: UNIX/Internet security. In *COMPSEC International 95. Twelfth World Conference on Computer Security, Audit and Control, 25-27 Oct. 1995, London, UK*. Elsevier, Oxford, UK.

Vigna, G. 1996. A topological characterization of TCP/IP security. Tech. Rep. TR-96.156, Dipartimento di Elettronica e Informazione, Politecnico di Milano, Piazza Leonardo da Vinci, 32, 20133 Milano, Italy. December. http://www2.elet.polimi.it/pub/data/Giovanni.Vigna/www_docs/pub/tr96156%.ps.gz Accessed 2002 Feb 20.

Vigna, G. 1997. A formal model for firewall testing. Unpublished paper. http://citeseer.nj.nec.com/279361.html Accessed 2002 Feb 20.

Watson, D., Smart, M., Malan, G., and Jahanian, F. 2001. Protocol scrubbing: network security through transparent flow modification. In *DARPA Information Survivability Conference & Exposition II, 2001. DISCEX '01. Proceedings*. Vol. 2. IEEE Computer Society, Los Alamitos, CA, USA, 108–118.

Weber, W. 1999. Firewall basics. In *4th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services. TELSIKS'99. Proceedings of Papers, 13-15 Oct. 1999, Nis, Yugoslavia*. Vol. 1. IEEE, Piscataway, NJ, USA, 300–305.

Wego Systems, Inc. 2001. Welcome to gnutella. http://gnutella.wego.com/ Accessed 2002 Feb 20.

Wool, A. 2001. Architecting the Lumeta firewall analyzer. In *Conference Proceedings: 10th USENIX Security Symposium*. USENIX Association, Berkleley, CA, USA, 85–97. http://www.usenix.org/events/sec01/full_papers/wool/wool.pdf Accessed 2002 Feb 20.

Yavwa, Y. 2000. The firewall technology. Student paper, available at: http://www.cs.uct.ac.za/courses/CS400W/NIS/resources.html Accessed 2002 Feb 20.

Zwicky, E. D., Cooper, S., and Chapman, D. B. 2000. *Building Internet Firewalls, 2nd Edition*. O'Reilly & Associates, 101 Morris St, Sebastopol, CA 95472 USA.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1076

Keysight, EX-1045
Page 1076

Received Month Year; revised Month Year; accepted Month Year



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1077

**Keysight, EX-1045**
**Page 1077**

Network Working Group
Request for Comments: 2616
Obsoletes: 2068
Category: Standards Track

R. Fielding
UC Irvine
J. Gettys
Compaq/W3C
J. Mogul
Compaq
H. Frystyk
W3C/MIT
L. Masinter
Xerox
P. Leach
Microsoft
T. Berners-Lee
W3C/MIT

June 1999

# Hypertext Transfer Protocol -- HTTP/1.1

## Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

## Abstract

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers [47]. A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred.

HTTP has been in use by the World-Wide Web global information initiative since 1990. This specification defines the protocol referred to as "HTTP/1.1", and is an update to RFC 2068 [33].



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

Table of Contents

1 Introduction .....7
1.1 Purpose.....7
1.2 Requirements .....8
1.3 Terminology .....8
1.4 Overall Operation .....12
2 Notational Conventions and Generic Grammar .....14
2.1 Augmented BNF .....14
2.2 Basic Rules .....15
3 Protocol Parameters .....17
3.1 HTTP Version .....17
3.2 Uniform Resource Identifiers .....18
3.2.1 General Syntax .....19
3.2.2 http URL .....19
3.2.3 URI Comparison .....20
3.3 Date/Time Formats .....20
3.3.1 Full Date .....20
3.3.2 Delta Seconds .....21
3.4 Character Sets .....21
3.4.1 Missing Charset .....22
3.5 Content Codings .....23
3.6 Transfer Codings .....24
3.6.1 Chunked Transfer Coding .....25
3.7 Media Types .....26
3.7.1 Canonicalization and Text Defaults .....27
3.7.2 Multipart Types .....27
3.8 Product Tokens .....28
3.9 Quality Values .....29
3.10 Language Tags .....29
3.11 Entity Tags .....30
3.12 Range Units .....30
4 HTTP Message .....31
4.1 Message Types .....31
4.2 Message Headers .....31
4.3 Message Body .....32
4.4 Message Length .....33
4.5 General Header Fields .....34
5 Request .....35
5.1 Request-Line .....35
5.1.1 Method .....36
5.1.2 Request-URI .....36
5.2 The Resource Identified by a Request .....38
5.3 Request Header Fields .....38
6 Response .....39
6.1 Status-Line .....39
6.1.1 Status Code and Reason Phrase .....39
6.2 Response Header Fields .....41





7 Entity .....42
7.1 Entity Header Fields .....42
7.2 Entity Body .....43
7.2.1 Type .....43
7.2.2 Entity Length .....43
8 Connections .....44
8.1 Persistent Connections .....44

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

8.1.1 Purpose ....44
8.1.2 Overall Operation ....45
8.1.3 Proxy Servers ....46
8.1.4 Practical Considerations ....46
8.2 Message Transmission Requirements ....47
8.2.1 Persistent Connections and Flow Control ....47
8.2.2 Monitoring Connections for Error Status Messages ....48
8.2.3 Use of the 100 (Continue) Status ....48
8.2.4 Client Behavior if Server Prematurely Closes Connection ..50
9 Method Definitions ....51
9.1 Safe and Idempotent Methods ....51
9.1.1 Safe Methods ....51
9.1.2 Idempotent Methods ....51
9.2 OPTIONS ....52
9.3 GET ....53
9.4 HEAD ....54
9.5 POST ....54
9.6 PUT ....55
9.7 DELETE ....56
9.8 TRACE ....56
9.9 CONNECT ....57
10 Status Code Definitions ....57
10.1 Informational 1xx ....57
10.1.1 100 Continue ....58
10.1.2 101 Switching Protocols ....58
10.2 Successful 2xx ....58
10.2.1 200 OK ....58
10.2.2 201 Created ....59
10.2.3 202 Accepted ....59
10.2.4 203 Non-Authoritative Information ....59
10.2.5 204 No Content ....60
10.2.6 205 Reset Content ....60
10.2.7 206 Partial Content ....60
10.3 Redirection 3xx ....61
10.3.1 300 Multiple Choices ....61
10.3.2 301 Moved Permanently ....62
10.3.3 302 Found ....62
10.3.4 303 See Other ....63
10.3.5 304 Not Modified ....63
10.3.6 305 Use Proxy ....64
10.3.7 306 (Unused) ....64





10.3.8 307 Temporary Redirect ....65
10.4 Client Error 4xx ....65
10.4.1 400 Bad Request ....65
10.4.2 401 Unauthorized ....66
10.4.3 402 Payment Required ....66
10.4.4 403 Forbidden ....66
10.4.5 404 Not Found ....66
10.4.6 405 Method Not Allowed ....66
10.4.7 406 Not Acceptable ....67
10.4.8 407 Proxy Authentication Required ....67
10.4.9 408 Request Timeout ....67
10.4.10 409 Conflict ....67
10.4.11 410 Gone ....68
10.4.12 411 Length Required ....68

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

10.4.13 412 Precondition Failed ....68
10.4.14 413 Request Entity Too Large ....69
10.4.15 414 Request-URI Too Long ....69
10.4.16 415 Unsupported Media Type ....69
10.4.17 416 Requested Range Not Satisfiable ....69
10.4.18 417 Expectation Failed ....70
10.5 Server Error 5xx ....70
10.5.1 500 Internal Server Error ....70
10.5.2 501 Not Implemented ....70
10.5.3 502 Bad Gateway ....70
10.5.4 503 Service Unavailable ....70
10.5.5 504 Gateway Timeout ....71
10.5.6 505 HTTP Version Not Supported ....71
11 Access Authentication ....71
12 Content Negotiation ....71
12.1 Server-driven Negotiation ....72
12.2 Agent-driven Negotiation ....73
12.3 Transparent Negotiation ....74
13 Caching in HTTP ....74
13.1.1 Cache Correctness ....75
13.1.2 Warnings ....76
13.1.3 Cache-control Mechanisms ....77
13.1.4 Explicit User Agent Warnings ....78
13.1.5 Exceptions to the Rules and Warnings ....78
13.1.6 Client-controlled Behavior ....79
13.2 Expiration Model ....79
13.2.1 Server-Specified Expiration ....79
13.2.2 Heuristic Expiration ....80
13.2.3 Age Calculations ....80
13.2.4 Expiration Calculations ....83
13.2.5 Disambiguating Expiration Values ....84
13.2.6 Disambiguating Multiple Responses ....84
13.3 Validation Model ....85
13.3.1 Last-Modified Dates ....86





13.3.2 Entity Tag Cache Validators ....86
13.3.3 Weak and Strong Validators ....86
13.3.4 Rules for When to Use Entity Tags and Last-Modified Dates.89
13.3.5 Non-validating Conditionals ....90
13.4 Response Cacheability ....91
13.5 Constructing Responses From Caches ....92
13.5.1 End-to-end and Hop-by-hop Headers ....92
13.5.2 Non-modifiable Headers ....92
13.5.3 Combining Headers ....94
13.5.4 Combining Byte Ranges ....95
13.6 Caching Negotiated Responses ....95
13.7 Shared and Non-Shared Caches ....96
13.8 Errors or Incomplete Response Cache Behavior ....97
13.9 Side Effects of GET and HEAD ....97
13.10 Invalidation After Updates or Deletions ....97
13.11 Write-Through Mandatory ....98
13.12 Cache Replacement ....99
13.13 History Lists ....99
14 Header Field Definitions ....100
14.1 Accept ....100
14.2 Accept-Charset ....102

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1081
Keysight, EX-1045
Page 1081

14.3 Accept-Encoding .....102
14.4 Accept-Language .....104
14.5 Accept-Ranges .....105
14.6 Age .....106
14.7 Allow .....106
14.8 Authorization .....107
14.9 Cache-Control .....108
14.9.1 What is Cacheable .....109
14.9.2 What May be Stored by Caches .....110
14.9.3 Modifications of the Basic Expiration Mechanism .....111
14.9.4 Cache Revalidation and Reload Controls .....113
14.9.5 No-Transform Directive .....115
14.9.6 Cache Control Extensions .....116
14.10 Connection .....117
14.11 Content-Encoding .....118
14.12 Content-Language .....118
14.13 Content-Length .....119
14.14 Content-Location .....120
14.15 Content-MD5 .....121
14.16 Content-Range .....122
14.17 Content-Type .....124
14.18 Date .....124
14.18.1 Clockless Origin Server Operation .....125
14.19 ETag .....126
14.20 Expect .....126
14.21 Expires .....127
14.22 From .....128



14.23 Host .....128
14.24 If-Match .....129
14.25 If-Modified-Since .....130
14.26 If-None-Match .....132
14.27 If-Range .....133
14.28 If-Unmodified-Since .....134
14.29 Last-Modified .....134
14.30 Location .....135
14.31 Max-Forwards .....136
14.32 Pragma .....136
14.33 Proxy-Authenticate .....137
14.34 Proxy-Authorization .....137
14.35 Range .....138
14.35.1 Byte Ranges .....138
14.35.2 Range Retrieval Requests .....139
14.36 Referer .....140
14.37 Retry-After .....141
14.38 Server .....141
14.39 TE .....142
14.40 Trailer .....143
14.41 Transfer-Encoding.....143
14.42 Upgrade .....144
14.43 User-Agent .....145
14.44 Vary .....145
14.45 Via .....146
14.46 Warning .....148
14.47 WWW-Authenticate .....150
15 Security Considerations .....150

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

15.1 Personal Information.....151
15.1.1 Abuse of Server Log Information .....151
15.1.2 Transfer of Sensitive Information .....151
15.1.3 Encoding Sensitive Information in URI's .....152
15.1.4 Privacy Issues Connected to Accept Headers .....152
15.2 Attacks Based On File and Path Names .....153
15.3 DNS Spoofing .....154
15.4 Location Headers and Spoofing .....154
15.5 Content-Disposition Issues .....154
15.6 Authentication Credentials and Idle Clients .....155
15.7 Proxies and Caching .....155
15.7.1 Denial of Service Attacks on Proxies.....156
16 Acknowledgments .....156
17 References .....158
18 Authors' Addresses .....162
19 Appendices .....164
19.1 Internet Media Type message/http and application/http .....164
19.2 Internet Media Type multipart/byteranges .....165
19.3 Tolerant Applications .....166
19.4 Differences Between HTTP Entities and RFC 2045 Entities ....167





19.4.1 MIME-Version .....167
19.4.2 Conversion to Canonical Form .....167
19.4.3 Conversion of Date Formats .....168
19.4.4 Introduction of Content-Encoding .....168
19.4.5 No Content-Transfer-Encoding .....168
19.4.6 Introduction of Transfer-Encoding .....169
19.4.7 MHTML and Line Length Limitations .....169
19.5 Additional Features .....169
19.5.1 Content-Disposition .....170
19.6 Compatibility with Previous Versions .....170
19.6.1 Changes from HTTP/1.0 .....171
19.6.2 Compatibility with HTTP/1.0 Persistent Connections .....172
19.6.3 Changes from RFC 2068 .....172
20 Index .....175
21 Full Copyright Statement .....176

## 1 Introduction

### 1.1 Purpose

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. HTTP has been in use by the World-Wide Web global information initiative since 1990. The first version of HTTP, referred to as HTTP/0.9, was a simple protocol for raw data transfer across the Internet. HTTP/1.0, as defined by RFC 1945 [6], improved the protocol by allowing messages to be in the format of MIME-like messages, containing metainformation about the data transferred and modifiers on the request/response semantics. However, HTTP/1.0 does not sufficiently take into consideration the effects of hierarchical proxies, caching, the need for persistent connections, or virtual hosts. In addition, the proliferation of incompletely-implemented applications calling themselves "HTTP/1.0" has necessitated a protocol version change in order for two communicating applications to determine each other's true capabilities.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

This specification defines the protocol referred to as "HTTP/1.1". This protocol includes more stringent requirements than HTTP/1.0 in order to ensure reliable implementation of its features.

Practical information systems require more functionality than simple retrieval, including search, front-end update, and annotation. HTTP allows an open-ended set of methods and headers that indicate the purpose of a request [47]. It builds on the discipline of reference provided by the Uniform Resource Identifier (URI) [3], as a location (URL) [4] or name (URN) [20], for indicating the resource to which a





method is to be applied. Messages are passed in a format similar to that used by Internet mail [9] as defined by the Multipurpose Internet Mail Extensions (MIME) [7].

HTTP is also used as a generic protocol for communication between user agents and proxies/gateways to other Internet systems, including those supported by the SMTP [16], NNTP [13], FTP [18], Gopher [2], and WAIS [10] protocols. In this way, HTTP allows basic hypermedia access to resources available from diverse applications.

## 1.2 Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [34].

An implementation is not compliant if it fails to satisfy one or more of the MUST or REQUIRED level requirements for the protocols it implements. An implementation that satisfies all the MUST or REQUIRED level and all the SHOULD level requirements for its protocols is said to be "unconditionally compliant"; one that satisfies all the MUST level requirements but not all the SHOULD level requirements for its protocols is said to be "conditionally compliant."

## 1.3 Terminology

This specification uses a number of terms to refer to the roles played by participants in, and objects of, the HTTP communication.

connection
: A transport layer virtual circuit established between two programs for the purpose of communication.

message
: The basic unit of HTTP communication, consisting of a structured sequence of octets matching the syntax defined in section 4 and transmitted via the connection.

request
: An HTTP request message, as defined in section 5.

response

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

An HTTP response message, as defined in section 6.



resource
A network data object or service that can be identified by a URI, as defined in section 3.2. Resources may be available in multiple representations (e.g. multiple languages, data formats, size, and resolutions) or vary in other ways.

entity
The information transferred as the payload of a request or response. An entity consists of metainformation in the form of entity-header fields and content in the form of an entity-body, as described in section 7.

representation
An entity included with a response that is subject to content negotiation, as described in section 12. There may exist multiple representations associated with a particular response status.

content negotiation
The mechanism for selecting the appropriate representation when servicing a request, as described in section 12. The representation of entities in any response can be negotiated (including error responses).

variant
A resource may have one, or more than one, representation(s) associated with it at any given instant. Each of these representations is termed a `varriant'. Use of the term `variant' does not necessarily imply that the resource is subject to content negotiation.

client
A program that establishes connections for the purpose of sending requests.

user agent
The client which initiates a request. These are often browsers, editors, spiders (web-traversing robots), or other end user tools.

server
An application program that accepts connections in order to service requests by sending back responses. Any given program may be capable of being both a client and a server; our use of these terms refers only to the role being performed by the program for a particular connection, rather than to the program's capabilities in general. Likewise, any server may act as an origin server, proxy, gateway, or tunnel, switching behavior based on the nature of each request.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

origin server
The server on which a given resource resides or is to be created.

proxy
An intermediary program which acts as both a server and a client for the purpose of making requests on behalf of other clients. Requests are serviced internally or by passing them on, with possible translation, to other servers. A proxy MUST implement both the client and server requirements of this specification. A "transparent proxy" is a proxy that does not modify the request or response beyond what is required for proxy authentication and identification. A "non-transparent proxy" is a proxy that modifies the request or response in order to provide some added service to the user agent, such as group annotation services, media type transformation, protocol reduction, or anonymity filtering. Except where either transparent or non-transparent behavior is explicitly stated, the HTTP proxy requirements apply to both types of proxies.

gateway
A server which acts as an intermediary for some other server. Unlike a proxy, a gateway receives requests as if it were the origin server for the requested resource; the requesting client may not be aware that it is communicating with a gateway.

tunnel
An intermediary program which is acting as a blind relay between two connections. Once active, a tunnel is not considered a party to the HTTP communication, though the tunnel may have been initiated by an HTTP request. The tunnel ceases to exist when both ends of the relayed connections are closed.

cache
A program's local store of response messages and the subsystem that controls its message storage, retrieval, and deletion. A cache stores cacheable responses in order to reduce the response time and network bandwidth consumption on future, equivalent requests. Any client or server may include a cache, though a cache cannot be used by a server that is acting as a tunnel.

cacheable
A response is cacheable if a cache is allowed to store a copy of the response message for use in answering subsequent requests. The rules for determining the cacheability of HTTP responses are defined in section 13. Even if a resource is cacheable, there may be additional constraints on whether a cache can use the cached copy for a particular request.



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

first-hand
A response is first-hand if it comes directly and without unnecessary delay from the origin server, perhaps via one or more proxies. A response is also first-hand if its validity has just been checked directly with the origin server.

explicit expiration time
The time at which the origin server intends that an entity should no longer be returned by a cache without further validation.

heuristic expiration time
An expiration time assigned by a cache when no explicit expiration time is available.

age
The age of a response is the time since it was sent by, or successfully validated with, the origin server.

freshness lifetime
The length of time between the generation of a response and its expiration time.

fresh
A response is fresh if its age has not yet exceeded its freshness lifetime.

stale
A response is stale if its age has passed its freshness lifetime.

semantically transparent
A cache behaves in a "semantically transparent" manner, with respect to a particular response, when its use affects neither the requesting client nor the origin server, except to improve performance. When a cache is semantically transparent, the client receives exactly the same response (except for hop-by-hop headers) that it would have received had its request been handled directly by the origin server.

validator
A protocol element (e.g., an entity tag or a Last-Modified time) that is used to find out whether a cache entry is an equivalent copy of an entity.

upstream/downstream
Upstream and downstream describe the flow of a message: all messages flow from upstream to downstream.





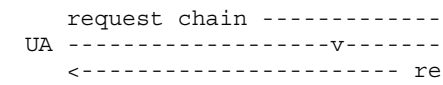
inbound/outbound
Inbound and outbound refer to the request and response paths for messages: "inbound" means "traveling toward the origin server", and "outbound" means "traveling toward the user agent"

## 1.4 Overall Operation

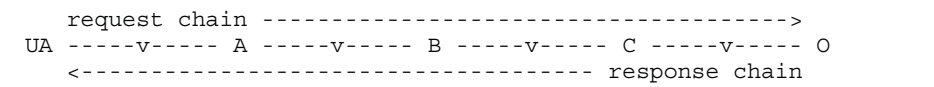
https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The HTTP protocol is a request/response protocol. A client sends a request to the server in the form of a request method, URI, and protocol version, followed by a MIME-like message containing request modifiers, client information, and possible body content over a connection with a server. The server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity metainformation, and possible entity-body content. The relationship between HTTP and MIME is described in appendix 19.4.

Most HTTP communication is initiated by a user agent and consists of a request to be applied to a resource on some origin server. In the simplest case, this may be accomplished via a single connection (v) between the user agent (UA) and the origin server (O).




A more complicated situation occurs when one or more intermediaries are present in the request/response chain. There are three common forms of intermediary: proxy, gateway, and tunnel. A proxy is a forwarding agent, receiving requests for a URI in its absolute form, rewriting all or part of the message, and forwarding the reformatted request toward the server identified by the URI. A gateway is a receiving agent, acting as a layer above some other server(s) and, if necessary, translating the requests to the underlying server's protocol. A tunnel acts as a relay point between two connections without changing the messages; tunnels are used when the communication needs to pass through an intermediary (such as a firewall) even when the intermediary cannot understand the contents of the messages.




The figure above shows three intermediaries (A, B, and C) between the user agent and origin server. A request or response message that travels the whole chain will pass through four separate connections. This distinction is important because some HTTP communication options





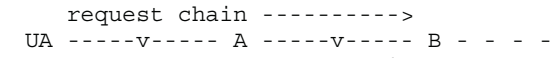
may apply only to the connection with the nearest, non-tunnel neighbor, only to the end-points of the chain, or to all connections along the chain. Although the diagram is linear, each participant may be engaged in multiple, simultaneous communications. For example, B may be receiving requests from many clients other than A, and/or forwarding requests to servers other than C, at the same time that it is handling A's request.

Any party to the communication which is not acting as a tunnel may employ an internal cache for handling requests. The effect of a cache is that the request/response chain is shortened if one of the participants along the chain has a cached response applicable to that request. The following illustrates the resulting chain if B has a cached copy of an earlier response from O (via C) for a request which

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1088
Keysight, EX-1045
Page 1088

has not been cached by UA or A.




Not all responses are usefully cacheable, and some requests may contain modifiers which place special requirements on cache behavior. HTTP requirements for cache behavior and cacheable responses are defined in section 13.

In fact, there are a wide variety of architectures and configurations of caches and proxies currently being experimented with or deployed across the World Wide Web. These systems include national hierarchies of proxy caches to save transoceanic bandwidth, systems that broadcast or multicast cache entries, organizations that distribute subsets of cached data via CD-ROM, and so on. HTTP systems are used in corporate intranets over high-bandwidth links, and for access via PDAs with low-power radio links and intermittent connectivity. The goal of HTTP/1.1 is to support the wide diversity of configurations already deployed while introducing protocol constructs that meet the needs of those who build web applications that require high reliability and, failing that, at least reliable indications of failure.

HTTP communication usually takes place over TCP/IP connections. The default port is TCP 80 [19], but other ports can be used. This does not preclude HTTP from being implemented on top of any other protocol on the Internet, or on other networks. HTTP only presumes a reliable transport; any protocol that provides such guarantees can be used; the mapping of the HTTP/1.1 request and response structures onto the transport data units of the protocol in question is outside the scope of this specification.





In HTTP/1.0, most implementations used a new connection for each request/response exchange. In HTTP/1.1, a connection may be used for one or more request/response exchanges, although connections may be closed for a variety of reasons (see section 8.1).

## 2 Notational Conventions and Generic Grammar

### 2.1 Augmented BNF

All of the mechanisms specified in this document are described in both prose and an augmented Backus-Naur Form (BNF) similar to that used by RFC 822 [9]. Implementors will need to be familiar with the notation in order to understand this specification. The augmented BNF includes the following constructs:

name = definition
The name of a rule is simply the name itself (without any enclosing "<" and ">") and is separated from its definition by the equal "=" character. White space is only significant in that indentation of continuation lines is used to indicate a rule definition that spans more than one line. Certain basic rules are

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1089
Keysight, EX-1045
Page 1089

in uppercase, such as SP, LWS, HT, CRLF, DIGIT, ALPHA, etc. Angle brackets are used within definitions whenever their presence will facilitate discerning the use of rule names.

"literal"
Quotation marks surround literal text. Unless stated otherwise, the text is case-insensitive.

rule1 | rule2
Elements separated by a bar ("|") are alternatives, e.g., "yes | no" will accept yes or no.

(rule1 rule2)
Elements enclosed in parentheses are treated as a single element. Thus, "(elem (foo | bar) elem)" allows the token sequences "elem foo elem" and "elem bar elem".

*rule
The character "*" preceding an element indicates repetition. The full form is "<n>*<m>element" indicating at least <n> and at most <m> occurrences of element. Default values are 0 and infinity so that "*(element)" allows any number, including zero; "1*element" requires at least one; and "1*2element" allows one or two.

[rule]
Square brackets enclose optional elements; "[foo bar]" is equivalent to "*1(foo bar)".





N rule
Specific repetition: "<n>(element)" is equivalent to "<n>*<n>(element)"; that is, exactly <n> occurrences of (element). Thus 2DIGIT is a 2-digit number, and 3ALPHA is a string of three alphabetic characters.

#rule
A construct "#" is defined, similar to "*", for defining lists of elements. The full form is "<n>#<m>element" indicating at least <n> and at most <m> elements, each separated by one or more commas (",") and OPTIONAL linear white space (LWS). This makes the usual form of lists very easy; a rule such as

```
( *LWS element *( *LWS "," *LWS element ))
```

can be shown as

```
1#element
```

Wherever this construct is used, null elements are allowed, but do not contribute to the count of elements present. That is, "(element), , (element) " is permitted, but counts as only two elements. Therefore, where at least one element is required, at least one non-null element MUST be present. Default values are 0 and infinity so that "#element" allows any number, including zero; "1#element" requires at least one; and "1#2element" allows one or two.

; comment
A semi-colon, set off some distance to the right of rule text, starts a comment that continues to the end of line. This is a simple way of including useful notes in parallel with the

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1090
Keysight, EX-1045
Page 1090

specifications.

implied *LWS
The grammar described by this specification is word-based. Except where noted otherwise, linear white space (LWS) can be included between any two adjacent words (token or quoted-string), and between adjacent words and separators, without changing the interpretation of a field. At least one delimiter (LWS and/or

separators) MUST exist between any two tokens (for the definition of "token" below), since they would otherwise be interpreted as a single token.

## 2.2 Basic Rules

The following rules are used throughout this specification to describe basic parsing constructs. The US-ASCII coded character set is defined by ANSI X3.4-1986 [21].





```
OCTET          = <any 8-bit sequence of data>
CHAR           = <any US-ASCII character (octets 0 - 127)>
UPALPHA        = <any US-ASCII uppercase letter "A".."Z">
LOALPHA        = <any US-ASCII lowercase letter "a".."z">
ALPHA          = UPALPHA | LOALPHA
DIGIT          = <any US-ASCII digit "0".."9">
CTL            = <any US-ASCII control character
                  (octets 0 - 31) and DEL (127)>
CR             = <US-ASCII CR, carriage return (13)>
LF             = <US-ASCII LF, linefeed (10)>
SP             = <US-ASCII SP, space (32)>
HT             = <US-ASCII HT, horizontal-tab (9)>
<">            = <US-ASCII double-quote mark (34)>
```

HTTP/1.1 defines the sequence CR LF as the end-of-line marker for all protocol elements except the entity-body (see appendix 19.3 for tolerant applications). The end-of-line marker within an entity-body is defined by its associated media type, as described in section 3.7.

```
CRLF           = CR LF
```

HTTP/1.1 header field values can be folded onto multiple lines if the continuation line begins with a space or horizontal tab. All linear white space, including folding, has the same semantics as SP. A recipient MAY replace any linear white space with a single SP before interpreting the field value or forwarding the message downstream.

```
LWS            = [CRLF] 1*( SP | HT )
```

The TEXT rule is only used for descriptive field contents and values that are not intended to be interpreted by the message parser. Words of *TEXT MAY contain characters from character sets other than ISO-8859-1 [22] only when encoded according to the rules of RFC 2047 [14].

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

```
    TEXT           = <any OCTET except CTLs,
                     but including LWS>
```

A CRLF is allowed in the definition of TEXT only as part of a header field continuation. It is expected that the folding LWS will be replaced with a single SP before interpretation of the TEXT value.

Hexadecimal numeric characters are used in several protocol elements.

```
    HEX            = "A" | "B" | "C" | "D" | "E" | "F"
                   | "a" | "b" | "c" | "d" | "e" | "f" | DIGIT
```





Many HTTP/1.1 header field values consist of words separated by LWS or special characters. These special characters MUST be in a quoted string to be used within a parameter value (as defined in section 3.6).

```
    token          = 1*<any CHAR except CTLs or separators>
    separators     = "(" | ")" | "<" | ">" | "@"
                   | "," | ";" | ":" | "\" | <">
                   | "/" | "[" | "]" | "?" | "="
                   | "{" | "}" | SP | HT
```

Comments can be included in some HTTP header fields by surrounding the comment text with parentheses. Comments are only allowed in fields containing "comment" as part of their field value definition. In all other fields, parentheses are considered part of the field value.

```
    comment        = "(" *( ctext | quoted-pair | comment ) ")"
    ctext          = <any TEXT excluding "(" and ")">
```

A string of text is parsed as a single word if it is quoted using double-quote marks.

```
    quoted-string  = ( <"> *(qdtext | quoted-pair ) <"> )
    qdtext         = <any TEXT except <">>
```

The backslash character ("\") MAY be used as a single-character quoting mechanism only within quoted-string and comment constructs.

```
    quoted-pair    = "\" CHAR
```

## 3 Protocol Parameters

### 3.1 HTTP Version

HTTP uses a "<major>.<minor>" numbering scheme to indicate versions of the protocol. The protocol versioning policy is intended to allow the sender to indicate the format of a message and its capacity for understanding further HTTP communication, rather than the features obtained via that communication. No change is made to the version number for the addition of message components which do not affect communication behavior or which only add to extensible field values.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The <minor> number is incremented when the changes made to the protocol add features which do not change the general message parsing algorithm, but which may add to the message semantics and imply additional capabilities of the sender. The <major> number is incremented when the format of a message within the protocol is changed. See RFC 2145 [36] for a fuller explanation.





The version of an HTTP message is indicated by an HTTP-Version field in the first line of the message.

```
HTTP-Version   = "HTTP" "/" 1*DIGIT "." 1*DIGIT
```

Note that the major and minor numbers MUST be treated as separate integers and that each MAY be incremented higher than a single digit. Thus, HTTP/2.4 is a lower version than HTTP/2.13, which in turn is lower than HTTP/12.3. Leading zeros MUST be ignored by recipients and MUST NOT be sent.

An application that sends a request or response message that includes HTTP-Version of "HTTP/1.1" MUST be at least conditionally compliant with this specification. Applications that are at least conditionally compliant with this specification SHOULD use an HTTP-Version of "HTTP/1.1" in their messages, and MUST do so for any message that is not compatible with HTTP/1.0. For more details on when to send specific HTTP-Version values, see RFC 2145 [36].

The HTTP version of an application is the highest HTTP version for which the application is at least conditionally compliant.

Proxy and gateway applications need to be careful when forwarding messages in protocol versions different from that of the application. Since the protocol version indicates the protocol capability of the sender, a proxy/gateway MUST NOT send a message with a version indicator which is greater than its actual version. If a higher version request is received, the proxy/gateway MUST either downgrade the request version, or respond with an error, or switch to tunnel behavior.

Due to interoperability problems with HTTP/1.0 proxies discovered since the publication of RFC 2068[33], caching proxies MUST, gateways MAY, and tunnels MUST NOT upgrade the request to the highest version they support. The proxy/gateway's response to that request MUST be in the same major version as the request.

> Note: Converting between versions of HTTP may involve modification of header fields required or forbidden by the versions involved.

## 3.2 Uniform Resource Identifiers

URIs have been known by many names: WWW addresses, Universal Document Identifiers, Universal Resource Identifiers [3], and finally the combination of Uniform Resource Locators (URL) [4] and Names (URN) [20]. As far as HTTP is concerned, Uniform Resource Identifiers are simply formatted strings which identify--via name, location, or any other characteristic--a resource.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

### 3.2.1 General Syntax

URIs in HTTP can be represented in absolute form or relative to some known base URI [11], depending upon the context of their use. The two forms are differentiated by the fact that absolute URIs always begin with a scheme name followed by a colon. For definitive information on URL syntax and semantics, see "Uniform Resource Identifiers (URI): Generic Syntax and Semantics," RFC 2396 [42] (which replaces RFCs 1738 [4] and RFC 1808 [11]). This specification adopts the definitions of "URI-reference", "absoluteURI", "relativeURI", "port", "host","abs_path", "rel_path", and "authority" from that specification.

The HTTP protocol does not place any a priori limit on the length of a URI. Servers MUST be able to handle the URI of any resource they serve, and SHOULD be able to handle URIs of unbounded length if they provide GET-based forms that could generate such URIs. A server SHOULD return 414 (Request-URI Too Long) status if a URI is longer than the server can handle (see section 10.4.15).

> Note: Servers ought to be cautious about depending on URI lengths above 255 bytes, because some older client or proxy implementations might not properly support these lengths.

### 3.2.2 http URL

The "http" scheme is used to locate network resources via the HTTP protocol. This section defines the scheme-specific syntax and semantics for http URLs.

```
http_URL = "http:" "//" host [ ":" port ] [ abs_path [ "?" query ]]
```

If the port is empty or not given, port 80 is assumed. The semantics are that the identified resource is located at the server listening for TCP connections on that port of that host, and the Request-URI for the resource is abs_path (section 5.1.2). The use of IP addresses in URLs SHOULD be avoided whenever possible (see RFC 1900 [24]). If the abs_path is not present in the URL, it MUST be given as "/" when used as a Request-URI for a resource (section 5.1.2). If a proxy receives a host name which is not a fully qualified domain name, it MAY add its domain to the host name it received. If a proxy receives a fully qualified domain name, the proxy MUST NOT change the host name.



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

### 3.2.3 URI Comparison

When comparing two URIs to decide if they match or not, a client SHOULD use a case-sensitive octet-by-octet comparison of the entire URIs, with these exceptions:

- A port that is empty or not given is equivalent to the default port for that URI-reference;

- Comparisons of host names MUST be case-insensitive;

- Comparisons of scheme names MUST be case-insensitive;

- An empty abs_path is equivalent to an abs_path of "/".

Characters other than those in the "reserved" and "unsafe" sets (see RFC 2396 [42]) are equivalent to their ""%" HEX HEX" encoding.

For example, the following three URIs are equivalent:

```
http://abc.com:80/~smith/home.html
http://ABC.com/%7Esmith/home.html
http://ABC.com:/%7esmith/home.html
```

## 3.3 Date/Time Formats

### 3.3.1 Full Date

HTTP applications have historically allowed three different formats for the representation of date/time stamps:

```
Sun, 06 Nov 1994 08:49:37 GMT  ; RFC 822, updated by RFC 1123
Sunday, 06-Nov-94 08:49:37 GMT ; RFC 850, obsoleted by RFC 1036
Sun Nov  6 08:49:37 1994       ; ANSI C's asctime() format
```

The first format is preferred as an Internet standard and represents a fixed-length subset of that defined by RFC 1123 [8] (an update to RFC 822 [9]). The second format is in common use, but is based on the obsolete RFC 850 [12] date format and lacks a four-digit year. HTTP/1.1 clients and servers that parse the date value MUST accept all three formats (for compatibility with HTTP/1.0), though they MUST only generate the RFC 1123 format for representing HTTP-date values in header fields. See section 19.3 for further information.

Note: Recipients of date values are encouraged to be robust in accepting date values that may have been sent by non-HTTP applications, as is sometimes the case when retrieving or posting messages via proxies/gateways to SMTP or NNTP.





All HTTP date/time stamps MUST be represented in Greenwich Mean Time (GMT), without exception. For the purposes of HTTP, GMT is exactly equal to UTC (Coordinated Universal Time). This is indicated in the first two formats by the inclusion of "GMT" as the three-letter abbreviation for time zone, and MUST be assumed when reading the asctime format. HTTP-date is case sensitive and MUST NOT include additional LWS beyond that specifically included as SP in the

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

grammar.

```
    HTTP-date    = rfc1123-date | rfc850-date | asctime-date
    rfc1123-date = wkday "," SP date1 SP time SP "GMT"
    rfc850-date  = weekday "," SP date2 SP time SP "GMT"
    asctime-date = wkday SP date3 SP time SP 4DIGIT
    date1        = 2DIGIT SP month SP 4DIGIT
                   ; day month year (e.g., 02 Jun 1982)
    date2        = 2DIGIT "-" month "-" 2DIGIT
                   ; day-month-year (e.g., 02-Jun-82)
    date3        = month SP ( 2DIGIT | ( SP 1DIGIT ))
                   ; month day (e.g., Jun  2)
    time         = 2DIGIT ":" 2DIGIT ":" 2DIGIT
                   ; 00:00:00 - 23:59:59
    wkday        = "Mon" | "Tue" | "Wed"
                 | "Thu" | "Fri" | "Sat" | "Sun"
    weekday      = "Monday" | "Tuesday" | "Wednesday"
                 | "Thursday" | "Friday" | "Saturday" | "Sunday"
    month        = "Jan" | "Feb" | "Mar" | "Apr"
                 | "May" | "Jun" | "Jul" | "Aug"
                 | "Sep" | "Oct" | "Nov" | "Dec"
```

Note: HTTP requirements for the date/time stamp format apply only to their usage within the protocol stream. Clients and servers are not required to use these formats for user presentation, request logging, etc.

### 3.3.2 Delta Seconds

Some HTTP header fields allow a time value to be specified as an integer number of seconds, represented in decimal, after the time that the message was received.

```
    delta-seconds  = 1*DIGIT
```

## 3.4 Character Sets

HTTP uses the same definition of the term "character set" as that described for MIME:





The term "character set" is used in this document to refer to a method used with one or more tables to convert a sequence of octets into a sequence of characters. Note that unconditional conversion in the other direction is not required, in that not all characters may be available in a given character set and a character set may provide more than one sequence of octets to represent a particular character. This definition is intended to allow various kinds of character encoding, from simple single-table mappings such as US-ASCII to complex table switching methods such as those that use ISO-2022's techniques. However, the definition associated with a MIME character set name MUST fully specify the mapping to be performed from octets to characters. In particular, use of external profiling information to determine the exact mapping is not permitted.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

Note: This use of the term "character set" is more commonly referred to as a "character encoding." However, since HTTP and MIME share the same registry, it is important that the terminology also be shared.

HTTP character sets are identified by case-insensitive tokens. The complete set of tokens is defined by the IANA Character Set registry [19].

```
    charset = token
```

Although HTTP allows an arbitrary token to be used as a charset value, any token that has a predefined value within the IANA Character Set registry [19] MUST represent the character set defined by that registry. Applications SHOULD limit their use of character sets to those defined by the IANA registry.

Implementors should be aware of IETF character set requirements [38] [41].

### 3.4.1 Missing Charset

Some HTTP/1.0 software has interpreted a Content-Type header without charset parameter incorrectly to mean "recipient should guess." Senders wishing to defeat this behavior MAY include a charset parameter even when the charset is ISO-8859-1 and SHOULD do so when it is known that it will not confuse the recipient.

Unfortunately, some older HTTP/1.0 clients did not deal properly with an explicit charset parameter. HTTP/1.1 recipients MUST respect the charset label provided by the sender; and those user agents that have a provision to "guess" a charset MUST use the charset from the





content-type field if they support that charset, rather than the recipient's preference, when initially displaying a document. See section 3.7.1.

## 3.5 Content Codings

Content coding values indicate an encoding transformation that has been or can be applied to an entity. Content codings are primarily used to allow a document to be compressed or otherwise usefully transformed without losing the identity of its underlying media type and without loss of information. Frequently, the entity is stored in coded form, transmitted directly, and only decoded by the recipient.

```
    content-coding   = token
```

All content-coding values are case-insensitive. HTTP/1.1 uses content-coding values in the Accept-Encoding (section 14.3) and Content-Encoding (section 14.11) header fields. Although the value describes the content-coding, what is more important is that it indicates what decoding mechanism will be required to remove the encoding.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The Internet Assigned Numbers Authority (IANA) acts as a registry for content-coding value tokens. Initially, the registry contains the following tokens:

gzip An encoding format produced by the file compression program "gzip" (GNU zip) as described in RFC 1952 [25]. This format is a Lempel-Ziv coding (LZ77) with a 32 bit CRC.

compress
The encoding format produced by the common UNIX file compression program "compress". This format is an adaptive Lempel-Ziv-Welch coding (LZW).

Use of program names for the identification of encoding formats is not desirable and is discouraged for future encodings. Their use here is representative of historical practice, not good design. For compatibility with previous implementations of HTTP, applications SHOULD consider "x-gzip" and "x-compress" to be equivalent to "gzip" and "compress" respectively.

deflate
The "zlib" format defined in RFC 1950 [31] in combination with the "deflate" compression mechanism described in RFC 1951 [29].





identity
The default (identity) encoding; the use of no transformation whatsoever. This content-coding is used only in the Accept-Encoding header, and SHOULD NOT be used in the Content-Encoding header.

New content-coding value tokens SHOULD be registered; to allow interoperability between clients and servers, specifications of the content coding algorithms needed to implement a new value SHOULD be publicly available and adequate for independent implementation, and conform to the purpose of content coding defined in this section.

### 3.6 Transfer Codings

Transfer-coding values are used to indicate an encoding transformation that has been, can be, or may need to be applied to an entity-body in order to ensure "safe transport" through the network. This differs from a content coding in that the transfer-coding is a property of the message, not of the original entity.

```
    transfer-coding         = "chunked" | transfer-extension
    transfer-extension      = token *( ";" parameter )
```

Parameters are in the form of attribute/value pairs.

```
    parameter               = attribute "=" value
    attribute               = token
    value                   = token | quoted-string
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

All transfer-coding values are case-insensitive. HTTP/1.1 uses transfer-coding values in the TE header field (section 14.39) and in the Transfer-Encoding header field (section 14.41).

Whenever a transfer-coding is applied to a message-body, the set of transfer-codings MUST include "chunked", unless the message is terminated by closing the connection. When the "chunked" transfer-coding is used, it MUST be the last transfer-coding applied to the message-body. The "chunked" transfer-coding MUST NOT be applied more than once to a message-body. These rules allow the recipient to determine the transfer-length of the message (section 4.4).

Transfer-codings are analogous to the Content-Transfer-Encoding values of MIME [7], which were designed to enable safe transport of binary data over a 7-bit transport service. However, safe transport has a different focus for an 8bit-clean transfer protocol. In HTTP, the only unsafe characteristic of message-bodies is the difficulty in determining the exact body length (section 7.2.2), or the desire to encrypt data over a shared transport.





The Internet Assigned Numbers Authority (IANA) acts as a registry for transfer-coding value tokens. Initially, the registry contains the following tokens: "chunked" (section 3.6.1), "identity" (section 3.6.2), "gzip" (section 3.5), "compress" (section 3.5), and "deflate" (section 3.5).

New transfer-coding value tokens SHOULD be registered in the same way as new content-coding value tokens (section 3.5).

A server which receives an entity-body with a transfer-coding it does not understand SHOULD return 501 (Unimplemented), and close the connection. A server MUST NOT send transfer-codings to an HTTP/1.0 client.

### 3.6.1 Chunked Transfer Coding

The chunked encoding modifies the body of a message in order to transfer it as a series of chunks, each with its own size indicator, followed by an OPTIONAL trailer containing entity-header fields. This allows dynamically produced content to be transferred along with the information necessary for the recipient to verify that it has received the full message.

```
    Chunked-Body   = *chunk
                     last-chunk
                     trailer
                     CRLF

    chunk          = chunk-size [ chunk-extension ] CRLF
                     chunk-data CRLF
    chunk-size     = 1*HEX
    last-chunk     = 1*("0") [ chunk-extension ] CRLF

    chunk-extension= *( ";" chunk-ext-name [ "=" chunk-ext-val ] )
    chunk-ext-name = token
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

```
    chunk-ext-val  = token | quoted-string
    chunk-data     = chunk-size(OCTET)
    trailer        = *(entity-header CRLF)
```

The chunk-size field is a string of hex digits indicating the size of the chunk. The chunked encoding is ended by any chunk whose size is zero, followed by the trailer, which is terminated by an empty line.

The trailer allows the sender to include additional HTTP header fields at the end of the message. The Trailer header field can be used to indicate which header fields are included in a trailer (see section 14.40).





A server using chunked transfer-coding in a response MUST NOT use the trailer for any header fields unless at least one of the following is true:

a) the request included a TE header field that indicates "trailers" is acceptable in the transfer-coding of the response, as described in section 14.39; or,

b) the server is the origin server for the response, the trailer fields consist entirely of optional metadata, and the recipient could use the message (in a manner acceptable to the origin server) without receiving this metadata. In other words, the origin server is willing to accept the possibility that the trailer fields might be silently discarded along the path to the client.

This requirement prevents an interoperability failure when the message is being received by an HTTP/1.1 (or later) proxy and forwarded to an HTTP/1.0 recipient. It avoids a situation where compliance with the protocol would have necessitated a possibly infinite buffer on the proxy.

An example process for decoding a Chunked-Body is presented in appendix 19.4.6.

All HTTP/1.1 applications MUST be able to receive and decode the "chunked" transfer-coding, and MUST ignore chunk-extension extensions they do not understand.

## 3.7 Media Types

HTTP uses Internet Media Types [17] in the Content-Type (section 14.17) and Accept (section 14.1) header fields in order to provide open and extensible data typing and type negotiation.

```
    media-type     = type "/" subtype *( ";" parameter )
    type           = token
    subtype        = token
```

Parameters MAY follow the type/subtype in the form of attribute/value pairs (as defined in section 3.6).

The type, subtype, and parameter attribute names are case-

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

insensitive. Parameter values might or might not be case-sensitive, depending on the semantics of the parameter name. Linear white space (LWS) MUST NOT be used between the type and subtype, nor between an attribute and its value. The presence or absence of a parameter might be significant to the processing of a media-type, depending on its definition within the media type registry.





Note that some older HTTP applications do not recognize media type parameters. When sending data to older HTTP applications, implementations SHOULD only use media type parameters when they are required by that type/subtype definition.

Media-type values are registered with the Internet Assigned Number Authority (IANA [19]). The media type registration process is outlined in RFC 1590 [17]. Use of non-registered media types is discouraged.

### 3.7.1 Canonicalization and Text Defaults

Internet media types are registered with a canonical form. An entity-body transferred via HTTP messages MUST be represented in the appropriate canonical form prior to its transmission except for "text" types, as defined in the next paragraph.

When in canonical form, media subtypes of the "text" type use CRLF as the text line break. HTTP relaxes this requirement and allows the transport of text media with plain CR or LF alone representing a line break when it is done consistently for an entire entity-body. HTTP applications MUST accept CRLF, bare CR, and bare LF as being representative of a line break in text media received via HTTP. In addition, if the text is represented in a character set that does not use octets 13 and 10 for CR and LF respectively, as is the case for some multi-byte character sets, HTTP allows the use of whatever octet sequences are defined by that character set to represent the equivalent of CR and LF for line breaks. This flexibility regarding line breaks applies only to text media in the entity-body; a bare CR or LF MUST NOT be substituted for CRLF within any of the HTTP control structures (such as header fields and multipart boundaries).

If an entity-body is encoded with a content-coding, the underlying data MUST be in a form defined above prior to being encoded.

The "charset" parameter is used with some media types to define the character set (section 3.4) of the data. When no explicit charset parameter is provided by the sender, media subtypes of the "text" type are defined to have a default charset value of "ISO-8859-1" when received via HTTP. Data in character sets other than "ISO-8859-1" or its subsets MUST be labeled with an appropriate charset value. See section 3.4.1 for compatibility problems.

### 3.7.2 Multipart Types

MIME provides for a number of "multipart" types -- encapsulations of one or more entities within a single message-body. All multipart types share a common syntax, as defined in section 5.1.1 of RFC 2046

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

[40], and MUST include a boundary parameter as part of the media type value. The message body is itself a protocol element and MUST therefore use only CRLF to represent line breaks between body-parts. Unlike in RFC 2046, the epilogue of any multipart message MUST be empty; HTTP applications MUST NOT transmit the epilogue (even if the original multipart contains an epilogue). These restrictions exist in order to preserve the self-delimiting nature of a multipart message-body, wherein the "end" of the message-body is indicated by the ending multipart boundary.

In general, HTTP treats a multipart message-body no differently than any other media type: strictly as payload. The one exception is the "multipart/byteranges" type (appendix 19.2) when it appears in a 206 (Partial Content) response, which will be interpreted by some HTTP caching mechanisms as described in sections 13.5.4 and 14.16. In all other cases, an HTTP user agent SHOULD follow the same or similar behavior as a MIME user agent would upon receipt of a multipart type. The MIME header fields within each body-part of a multipart message-body do not have any significance to HTTP beyond that defined by their MIME semantics.

In general, an HTTP user agent SHOULD follow the same or similar behavior as a MIME user agent would upon receipt of a multipart type. If an application receives an unrecognized multipart subtype, the application MUST treat it as being equivalent to "multipart/mixed".

> Note: The "multipart/form-data" type has been specifically defined for carrying form data suitable for processing via the POST request method, as described in RFC 1867 [15].

## 3.8 Product Tokens

Product tokens are used to allow communicating applications to identify themselves by software name and version. Most fields using product tokens also allow sub-products which form a significant part of the application to be listed, separated by white space. By convention, the products are listed in order of their significance for identifying the application.

```
product         = token ["/" product-version]
product-version = token
```

Examples:

```
User-Agent: CERN-LineMode/2.15 libwww/2.17b3
Server: Apache/0.8.4
```



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

Product tokens SHOULD be short and to the point. They MUST NOT be used for advertising or other non-essential information. Although any token character MAY appear in a product-version, this token SHOULD only be used for a version identifier (i.e., successive versions of the same product SHOULD only differ in the product-version portion of the product value).

### 3.9 Quality Values

HTTP content negotiation (section 12) uses short "floating point" numbers to indicate the relative importance ("weight") of various negotiable parameters. A weight is normalized to a real number in the range 0 through 1, where 0 is the minimum and 1 the maximum value. If a parameter has a quality value of 0, then content with this parameter is `not acceptable' for the client. HTTP/1.1 applications MUST NOT generate more than three digits after the decimal point. User configuration of these values SHOULD also be limited in this fashion.

```
    qvalue         = ( "0" [ "." 0*3DIGIT ] )
                   | ( "1" [ "." 0*3("0") ] )
```

"Quality values" is a misnomer, since these values merely represent relative degradation in desired quality.

### 3.10 Language Tags

A language tag identifies a natural language spoken, written, or otherwise conveyed by human beings for communication of information to other human beings. Computer languages are explicitly excluded. HTTP uses language tags within the Accept-Language and Content-Language fields.

The syntax and registry of HTTP language tags is the same as that defined by RFC 1766 [1]. In summary, a language tag is composed of 1 or more parts: A primary language tag and a possibly empty series of subtags:

```
     language-tag  = primary-tag *( "-" subtag )
     primary-tag   = 1*8ALPHA
     subtag        = 1*8ALPHA
```

White space is not allowed within the tag and all tags are case-insensitive. The name space of language tags is administered by the IANA. Example tags include:

```
    en, en-US, en-cockney, i-cherokee, x-pig-latin
```





where any two-letter primary-tag is an ISO-639 language abbreviation and any two-letter initial subtag is an ISO-3166 country code. (The last three tags above are not registered tags; all but the last are examples of tags which could be registered in future.)

### 3.11 Entity Tags

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1103
Keysight, EX-1045
Page 1103

Entity tags are used for comparing two or more entities from the same requested resource. HTTP/1.1 uses entity tags in the ETag (section 14.19), If-Match (section 14.24), If-None-Match (section 14.26), and If-Range (section 14.27) header fields. The definition of how they are used and compared as cache validators is in section 13.3.3. An entity tag consists of an opaque quoted string, possibly prefixed by a weakness indicator.

```
entity-tag = [ weak ] opaque-tag
weak       = "W/"
opaque-tag = quoted-string
```

A "strong entity tag" MAY be shared by two entities of a resource only if they are equivalent by octet equality.

A "weak entity tag," indicated by the "W/" prefix, MAY be shared by two entities of a resource only if the entities are equivalent and could be substituted for each other with no significant change in semantics. A weak entity tag can only be used for weak comparison.

An entity tag MUST be unique across all versions of all entities associated with a particular resource. A given entity tag value MAY be used for entities obtained by requests on different URIs. The use of the same entity tag value in conjunction with entities obtained by requests on different URIs does not imply the equivalence of those entities.

### 3.12 Range Units

HTTP/1.1 allows a client to request that only part (a range of) the response entity be included within the response. HTTP/1.1 uses range units in the Range (section 14.35) and Content-Range (section 14.16) header fields. An entity can be broken down into subranges according to various structural units.

```
range-unit       = bytes-unit | other-range-unit
bytes-unit       = "bytes"
other-range-unit = token
```

The only range unit defined by HTTP/1.1 is "bytes". HTTP/1.1 implementations MAY ignore ranges specified using other units.





HTTP/1.1 has been designed to allow implementations of applications that do not depend on knowledge of ranges.

## 4 HTTP Message

### 4.1 Message Types

HTTP messages consist of requests from client to server and responses from server to client.

```
HTTP-message   = Request | Response     ; HTTP/1.1 messages
```

Request (section 5) and Response (section 6) messages use the generic message format of RFC 822 [9] for transferring entities (the payload

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1104
Keysight, EX-1045
Page 1104

of the message). Both types of message consist of a start-line, zero or more header fields (also known as "headers"), an empty line (i.e., a line with nothing preceding the CRLF) indicating the end of the header fields, and possibly a message-body.

```
     generic-message = start-line
                       *(message-header CRLF)
                       CRLF
                       [ message-body ]
     start-line      = Request-Line | Status-Line
```

In the interest of robustness, servers SHOULD ignore any empty line(s) received where a Request-Line is expected. In other words, if the server is reading the protocol stream at the beginning of a message and receives a CRLF first, it should ignore the CRLF.

Certain buggy HTTP/1.0 client implementations generate extra CRLF's after a POST request. To restate what is explicitly forbidden by the BNF, an HTTP/1.1 client MUST NOT preface or follow a request with an extra CRLF.

## 4.2 Message Headers

HTTP header fields, which include general-header (section 4.5), request-header (section 5.3), response-header (section 6.2), and entity-header (section 7.1) fields, follow the same generic format as that given in Section 3.1 of RFC 822 [9]. Each header field consists of a name followed by a colon (":") and the field value. Field names are case-insensitive. The field value MAY be preceded by any amount of LWS, though a single SP is preferred. Header fields can be extended over multiple lines by preceding each extra line with at least one SP or HT. Applications ought to follow "common form", where one is known or indicated, when generating HTTP constructs, since there might exist some implementations that fail to accept anything





beyond the common forms.

```
    message-header = field-name ":" [ field-value ]
    field-name     = token
    field-value    = *( field-content | LWS )
    field-content  = <the OCTETs making up the field-value
                     and consisting of either *TEXT or combinations
                     of token, separators, and quoted-string>
```

The field-content does not include any leading or trailing LWS: linear white space occurring before the first non-whitespace character of the field-value or after the last non-whitespace character of the field-value. Such leading or trailing LWS MAY be removed without changing the semantics of the field value. Any LWS that occurs between field-content MAY be replaced with a single SP before interpreting the field value or forwarding the message downstream.

The order in which header fields with differing field names are received is not significant. However, it is "good practice" to send general-header fields first, followed by request-header or response-

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

header fields, and ending with the entity-header fields.

Multiple message-header fields with the same field-name MAY be present in a message if and only if the entire field-value for that header field is defined as a comma-separated list [i.e., #(values)]. It MUST be possible to combine the multiple header fields into one "field-name: field-value" pair, without changing the semantics of the message, by appending each subsequent field-value to the first, each separated by a comma. The order in which header fields with the same field-name are received is therefore significant to the interpretation of the combined field value, and thus a proxy MUST NOT change the order of these field values when a message is forwarded.

## 4.3 Message Body

The message-body (if any) of an HTTP message is used to carry the entity-body associated with the request or response. The message-body differs from the entity-body only when a transfer-coding has been applied, as indicated by the Transfer-Encoding header field (section 14.41).

```
    message-body = entity-body
                 | <entity-body encoded as per Transfer-Encoding>
```

Transfer-Encoding MUST be used to indicate any transfer-codings applied by an application to ensure safe and proper transfer of the message. Transfer-Encoding is a property of the message, not of the





entity, and thus MAY be added or removed by any application along the request/response chain. (However, section 3.6 places restrictions on when certain transfer-codings may be used.)

The rules for when a message-body is allowed in a message differ for requests and responses.

The presence of a message-body in a request is signaled by the inclusion of a Content-Length or Transfer-Encoding header field in the request's message-headers. A message-body MUST NOT be included in a request if the specification of the request method (section 5.1.1) does not allow sending an entity-body in requests. A server SHOULD read and forward a message-body on any request; if the request method does not include defined semantics for an entity-body, then the message-body SHOULD be ignored when handling the request.

For response messages, whether or not a message-body is included with a message is dependent on both the request method and the response status code (section 6.1.1). All responses to the HEAD request method MUST NOT include a message-body, even though the presence of entity-header fields might lead one to believe they do. All 1xx (informational), 204 (no content), and 304 (not modified) responses MUST NOT include a message-body. All other responses do include a message-body, although it MAY be of zero length.

## 4.4 Message Length

The transfer-length of a message is the length of the message-body as

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

it appears in the message; that is, after any transfer-codings have been applied. When a message-body is included with a message, the transfer-length of that body is determined by one of the following (in order of precedence):

1. Any response message which "MUST NOT" include a message-body (such as the 1xx, 204, and 304 responses and any response to a HEAD request) is always terminated by the first empty line after the header fields, regardless of the entity-header fields present in the message.

2. If a Transfer-Encoding header field (section 14.41) is present and has any value other than "identity", then the transfer-length is defined by use of the "chunked" transfer-coding (section 3.6), unless the message is terminated by closing the connection.

3. If a Content-Length header field (section 14.13) is present, its decimal value in OCTETs represents both the entity-length and the transfer-length. The Content-Length header field MUST NOT be sent if these two lengths are different (i.e., if a Transfer-Encoding





header field is present). If a message is received with both a Transfer-Encoding header field and a Content-Length header field, the latter MUST be ignored.

4. If the message uses the media type "multipart/byteranges", and the ransfer-length is not otherwise specified, then this self-elimiting media type defines the transfer-length. This media type UST NOT be used unless the sender knows that the recipient can arse it; the presence in a request of a Range header with ultiple byte-range specifiers from a 1.1 client implies that the lient can parse multipart/byteranges responses.

   A range header might be forwarded by a 1.0 proxy that does not understand multipart/byteranges; in this case the server MUST delimit the message using methods defined in items 1,3 or 5 of this section.

5. By the server closing the connection. (Closing the connection cannot be used to indicate the end of a request body, since that would leave no possibility for the server to send back a response.)

For compatibility with HTTP/1.0 applications, HTTP/1.1 requests containing a message-body MUST include a valid Content-Length header field unless the server is known to be HTTP/1.1 compliant. If a request contains a message-body and a Content-Length is not given, the server SHOULD respond with 400 (bad request) if it cannot determine the length of the message, or with 411 (length required) if it wishes to insist on receiving a valid Content-Length.

All HTTP/1.1 applications that receive entities MUST accept the "chunked" transfer-coding (section 3.6), thus allowing this mechanism to be used for messages when the message length cannot be determined in advance.

Messages MUST NOT include both a Content-Length header field and a

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1107
Keysight, EX-1045
Page 1107

non-identity transfer-coding. If the message does include a non-identity transfer-coding, the Content-Length MUST be ignored.

When a Content-Length is given in a message where a message-body is allowed, its field value MUST exactly match the number of OCTETs in the message-body. HTTP/1.1 user agents MUST notify the user when an invalid length is received and detected.

### 4.5 General Header Fields

There are a few header fields which have general applicability for both request and response messages, but which do not apply to the entity being transferred. These header fields apply only to the





message being transmitted.

```
    general-header = Cache-Control            ; Section 14.9
                   | Connection               ; Section 14.10
                   | Date                     ; Section 14.18
                   | Pragma                   ; Section 14.32
                   | Trailer                  ; Section 14.40
                   | Transfer-Encoding        ; Section 14.41
                   | Upgrade                  ; Section 14.42
                   | Via                      ; Section 14.45
                   | Warning                  ; Section 14.46
```

General-header field names can be extended reliably only in combination with a change in the protocol version. However, new or experimental header fields may be given the semantics of general header fields if all parties in the communication recognize them to be general-header fields. Unrecognized header fields are treated as entity-header fields.

## 5 Request

A request message from a client to a server includes, within the first line of that message, the method to be applied to the resource, the identifier of the resource, and the protocol version in use.

```
     Request       = Request-Line              ; Section 5.1
                     *(( general-header        ; Section 4.5
                      | request-header         ; Section 5.3
                      | entity-header ) CRLF)  ; Section 7.1
                     CRLF
                     [ message-body ]          ; Section 4.3
```

### 5.1 Request-Line

The Request-Line begins with a method token, followed by the Request-URI and the protocol version, and ending with CRLF. The elements are separated by SP characters. No CR or LF is allowed except in the final CRLF sequence.

```
     Request-Line   = Method SP Request-URI SP HTTP-Version CRLF
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

### 5.1.1 Method

The Method token indicates the method to be performed on the resource identified by the Request-URI. The method is case-sensitive.

```
    Method         = "OPTIONS"                ; Section 9.2
                   | "GET"                    ; Section 9.3
                   | "HEAD"                   ; Section 9.4
                   | "POST"                   ; Section 9.5
                   | "PUT"                    ; Section 9.6
                   | "DELETE"                 ; Section 9.7
                   | "TRACE"                  ; Section 9.8
                   | "CONNECT"                ; Section 9.9
                   | extension-method
    extension-method = token
```

The list of methods allowed by a resource can be specified in an Allow header field (section 14.7). The return code of the response always notifies the client whether a method is currently allowed on a resource, since the set of allowed methods can change dynamically. An origin server SHOULD return the status code 405 (Method Not Allowed) if the method is known by the origin server but not allowed for the requested resource, and 501 (Not Implemented) if the method is unrecognized or not implemented by the origin server. The methods GET and HEAD MUST be supported by all general-purpose servers. All other methods are OPTIONAL; however, if the above methods are implemented, they MUST be implemented with the same semantics as those specified in section 9.

### 5.1.2 Request-URI

The Request-URI is a Uniform Resource Identifier (section 3.2) and identifies the resource upon which to apply the request.

```
    Request-URI    = "*" | absoluteURI | abs_path | authority
```

The four options for Request-URI are dependent on the nature of the request. The asterisk "*" means that the request does not apply to a particular resource, but to the server itself, and is only allowed when the method used does not necessarily apply to a resource. One example would be

```
    OPTIONS * HTTP/1.1
```

The absoluteURI form is REQUIRED when the request is being made to a proxy. The proxy is requested to forward the request or service it from a valid cache, and return the response. Note that the proxy MAY forward the request on to another proxy or directly to the server

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

specified by the absoluteURI. In order to avoid request loops, a proxy MUST be able to recognize all of its server names, including any aliases, local variations, and the numeric IP address. An example Request-Line would be:

```
GET http://www.w3.org/pub/WWW/TheProject.html HTTP/1.1
```

To allow for transition to absoluteURIs in all requests in future versions of HTTP, all HTTP/1.1 servers MUST accept the absoluteURI form in requests, even though HTTP/1.1 clients will only generate them in requests to proxies.

The authority form is only used by the CONNECT method (section 9.9).

The most common form of Request-URI is that used to identify a resource on an origin server or gateway. In this case the absolute path of the URI MUST be transmitted (see section 3.2.1, abs_path) as the Request-URI, and the network location of the URI (authority) MUST be transmitted in a Host header field. For example, a client wishing to retrieve the resource above directly from the origin server would create a TCP connection to port 80 of the host "www.w3.org" and send the lines:

```
GET /pub/WWW/TheProject.html HTTP/1.1
Host: www.w3.org
```

followed by the remainder of the Request. Note that the absolute path cannot be empty; if none is present in the original URI, it MUST be given as "/" (the server root).

The Request-URI is transmitted in the format specified in section 3.2.1. If the Request-URI is encoded using the "% HEX HEX" encoding [42], the origin server MUST decode the Request-URI in order to properly interpret the request. Servers SHOULD respond to invalid Request-URIs with an appropriate status code.

A transparent proxy MUST NOT rewrite the "abs_path" part of the received Request-URI when forwarding it to the next inbound server, except as noted above to replace a null abs_path with "/".

> Note: The "no rewrite" rule prevents the proxy from changing the meaning of the request when the origin server is improperly using a non-reserved URI character for a reserved purpose. Implementors should be aware that some pre-HTTP/1.1 proxies have been known to rewrite the Request-URI.



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

## 5.2 The Resource Identified by a Request

The exact resource identified by an Internet request is determined by examining both the Request-URI and the Host header field.

An origin server that does not allow resources to differ by the requested host MAY ignore the Host header field value when determining the resource identified by an HTTP/1.1 request. (But see section 19.6.1.1 for other requirements on Host support in HTTP/1.1.)

An origin server that does differentiate resources based on the host requested (sometimes referred to as virtual hosts or vanity host names) MUST use the following rules for determining the requested resource on an HTTP/1.1 request:

1. If Request-URI is an absoluteURI, the host is part of the Request-URI. Any Host header field value in the request MUST be ignored.

2. If the Request-URI is not an absoluteURI, and the request includes a Host header field, the host is determined by the Host header field value.

3. If the host as determined by rule 1 or 2 is not a valid host on the server, the response MUST be a 400 (Bad Request) error message.

Recipients of an HTTP/1.0 request that lacks a Host header field MAY attempt to use heuristics (e.g., examination of the URI path for something unique to a particular host) in order to determine what exact resource is being requested.

## 5.3 Request Header Fields

The request-header fields allow the client to pass additional information about the request, and about the client itself, to the server. These fields act as request modifiers, with semantics equivalent to the parameters on a programming language method invocation.

```
    request-header = Accept                   ; Section 14.1
                   | Accept-Charset           ; Section 14.2
                   | Accept-Encoding          ; Section 14.3
                   | Accept-Language          ; Section 14.4
                   | Authorization            ; Section 14.8
                   | Expect                   ; Section 14.20
                   | From                     ; Section 14.22
                   | Host                     ; Section 14.23
                   | If-Match                 ; Section 14.24
```





```
                   | If-Modified-Since        ; Section 14.25
                   | If-None-Match           ; Section 14.26
                   | If-Range                ; Section 14.27
                   | If-Unmodified-Since     ; Section 14.28
                   | Max-Forwards            ; Section 14.31
                   | Proxy-Authorization     ; Section 14.34
                   | Range                   ; Section 14.35
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1111
Keysight, EX-1045
Page 1111

```
                    | Referer                  ; Section 14.36
                    | TE                       ; Section 14.39
                    | User-Agent               ; Section 14.43
```

Request-header field names can be extended reliably only in combination with a change in the protocol version. However, new or experimental header fields MAY be given the semantics of request-header fields if all parties in the communication recognize them to be request-header fields. Unrecognized header fields are treated as entity-header fields.

## 6 Response

After receiving and interpreting a request message, a server responds with an HTTP response message.

```
    Response      = Status-Line               ; Section 6.1
                    *(( general-header        ; Section 4.5
                     | response-header        ; Section 6.2
                     | entity-header ) CRLF)  ; Section 7.1
                    CRLF
                    [ message-body ]          ; Section 7.2
```

### 6.1 Status-Line

The first line of a Response message is the Status-Line, consisting of the protocol version followed by a numeric status code and its associated textual phrase, with each element separated by SP characters. No CR or LF is allowed except in the final CRLF sequence.

```
    Status-Line = HTTP-Version SP Status-Code SP Reason-Phrase CRLF
```

#### 6.1.1 Status Code and Reason Phrase

The Status-Code element is a 3-digit integer result code of the attempt to understand and satisfy the request. These codes are fully defined in section 10. The Reason-Phrase is intended to give a short textual description of the Status-Code. The Status-Code is intended for use by automata and the Reason-Phrase is intended for the human user. The client is not required to examine or display the Reason-Phrase.





The first digit of the Status-Code defines the class of response. The last two digits do not have any categorization role. There are 5 values for the first digit:

- 1xx: Informational - Request received, continuing process
- 2xx: Success - The action was successfully received, understood, and accepted
- 3xx: Redirection - Further action must be taken in order to complete the request
- 4xx: Client Error - The request contains bad syntax or cannot be fulfilled

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

```
   - 5xx: Server Error - The server failed to fulfill an apparently
     valid request

The individual values of the numeric status codes defined for
HTTP/1.1, and an example set of corresponding Reason-Phrase's, are
presented below. The reason phrases listed here are only
recommendations -- they MAY be replaced by local equivalents without
affecting the protocol.

   Status-Code    =
         "100"  ; Section 10.1.1: Continue
       | "101"  ; Section 10.1.2: Switching Protocols
       | "200"  ; Section 10.2.1: OK
       | "201"  ; Section 10.2.2: Created
       | "202"  ; Section 10.2.3: Accepted
       | "203"  ; Section 10.2.4: Non-Authoritative Information
       | "204"  ; Section 10.2.5: No Content
       | "205"  ; Section 10.2.6: Reset Content
       | "206"  ; Section 10.2.7: Partial Content
       | "300"  ; Section 10.3.1: Multiple Choices
       | "301"  ; Section 10.3.2: Moved Permanently
       | "302"  ; Section 10.3.3: Found
       | "303"  ; Section 10.3.4: See Other
       | "304"  ; Section 10.3.5: Not Modified
       | "305"  ; Section 10.3.6: Use Proxy
       | "307"  ; Section 10.3.8: Temporary Redirect
       | "400"  ; Section 10.4.1: Bad Request
       | "401"  ; Section 10.4.2: Unauthorized
       | "402"  ; Section 10.4.3: Payment Required
       | "403"  ; Section 10.4.4: Forbidden
       | "404"  ; Section 10.4.5: Not Found
       | "405"  ; Section 10.4.6: Method Not Allowed
       | "406"  ; Section 10.4.7: Not Acceptable
```





```
       | "407"  ; Section 10.4.8: Proxy Authentication Required
       | "408"  ; Section 10.4.9: Request Time-out
       | "409"  ; Section 10.4.10: Conflict
       | "410"  ; Section 10.4.11: Gone
       | "411"  ; Section 10.4.12: Length Required
       | "412"  ; Section 10.4.13: Precondition Failed
       | "413"  ; Section 10.4.14: Request Entity Too Large
       | "414"  ; Section 10.4.15: Request-URI Too Large
       | "415"  ; Section 10.4.16: Unsupported Media Type
       | "416"  ; Section 10.4.17: Requested range not satisfiable
       | "417"  ; Section 10.4.18: Expectation Failed
       | "500"  ; Section 10.5.1: Internal Server Error
       | "501"  ; Section 10.5.2: Not Implemented
       | "502"  ; Section 10.5.3: Bad Gateway
       | "503"  ; Section 10.5.4: Service Unavailable
       | "504"  ; Section 10.5.5: Gateway Time-out
       | "505"  ; Section 10.5.6: HTTP Version not supported
       | extension-code

   extension-code = 3DIGIT
   Reason-Phrase  = *<TEXT, excluding CR, LF>
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

HTTP status codes are extensible. HTTP applications are not required to understand the meaning of all registered status codes, though such understanding is obviously desirable. However, applications MUST understand the class of any status code, as indicated by the first digit, and treat any unrecognized response as being equivalent to the x00 status code of that class, with the exception that an unrecognized response MUST NOT be cached. For example, if an unrecognized status code of 431 is received by the client, it can safely assume that there was something wrong with its request and treat the response as if it had received a 400 status code. In such cases, user agents SHOULD present to the user the entity returned with the response, since that entity is likely to include human-readable information which will explain the unusual status.

### 6.2 Response Header Fields

The response-header fields allow the server to pass additional information about the response which cannot be placed in the Status-Line. These header fields give information about the server and about further access to the resource identified by the Request-URI.

```
    response-header = Accept-Ranges           ; Section 14.5
                    | Age                     ; Section 14.6
                    | ETag                    ; Section 14.19
                    | Location                ; Section 14.30
                    | Proxy-Authenticate      ; Section 14.33
```





```
                    | Retry-After             ; Section 14.37
                    | Server                  ; Section 14.38
                    | Vary                    ; Section 14.44
                    | WWW-Authenticate        ; Section 14.47
```

Response-header field names can be extended reliably only in combination with a change in the protocol version. However, new or experimental header fields MAY be given the semantics of response-header fields if all parties in the communication recognize them to be response-header fields. Unrecognized header fields are treated as entity-header fields.

## 7 Entity

Request and Response messages MAY transfer an entity if not otherwise restricted by the request method or response status code. An entity consists of entity-header fields and an entity-body, although some responses will only include the entity-headers.

In this section, both sender and recipient refer to either the client or the server, depending on who sends and who receives the entity.

### 7.1 Entity Header Fields

Entity-header fields define metainformation about the entity-body or, if no body is present, about the resource identified by the request. Some of this metainformation is OPTIONAL; some might be REQUIRED by portions of this specification.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

```
    entity-header  = Allow                    ; Section 14.7
                   | Content-Encoding         ; Section 14.11
                   | Content-Language         ; Section 14.12
                   | Content-Length           ; Section 14.13
                   | Content-Location         ; Section 14.14
                   | Content-MD5              ; Section 14.15
                   | Content-Range            ; Section 14.16
                   | Content-Type             ; Section 14.17
                   | Expires                  ; Section 14.21
                   | Last-Modified            ; Section 14.29
                   | extension-header

    extension-header = message-header
```

The extension-header mechanism allows additional entity-header fields to be defined without changing the protocol, but these fields cannot be assumed to be recognizable by the recipient. Unrecognized header fields SHOULD be ignored by the recipient and MUST be forwarded by transparent proxies.





## 7.2 Entity Body

The entity-body (if any) sent with an HTTP request or response is in a format and encoding defined by the entity-header fields.

```
    entity-body    = *OCTET
```

An entity-body is only present in a message when a message-body is present, as described in section 4.3. The entity-body is obtained from the message-body by decoding any Transfer-Encoding that might have been applied to ensure safe and proper transfer of the message.

### 7.2.1 Type

When an entity-body is included with a message, the data type of that body is determined via the header fields Content-Type and Content-Encoding. These define a two-layer, ordered encoding model:

```
    entity-body := Content-Encoding( Content-Type( data ) )
```

Content-Type specifies the media type of the underlying data. Content-Encoding may be used to indicate any additional content codings applied to the data, usually for the purpose of data compression, that are a property of the requested resource. There is no default encoding.

Any HTTP/1.1 message containing an entity-body SHOULD include a Content-Type header field defining the media type of that body. If and only if the media type is not given by a Content-Type field, the recipient MAY attempt to guess the media type via inspection of its content and/or the name extension(s) of the URI used to identify the resource. If the media type remains unknown, the recipient SHOULD treat it as type "application/octet-stream".

### 7.2.2 Entity Length

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The entity-length of a message is the length of the message-body before any transfer-codings have been applied. Section 4.4 defines how the transfer-length of a message-body is determined.





# 8 Connections

## 8.1 Persistent Connections

### 8.1.1 Purpose

Prior to persistent connections, a separate TCP connection was established to fetch each URL, increasing the load on HTTP servers and causing congestion on the Internet. The use of inline images and other associated data often require a client to make multiple requests of the same server in a short amount of time. Analysis of these performance problems and results from a prototype implementation are available [26] [30]. Implementation experience and measurements of actual HTTP/1.1 (RFC 2068) implementations show good results [39]. Alternatives have also been explored, for example, T/TCP [27].

Persistent HTTP connections have a number of advantages:

- By opening and closing fewer TCP connections, CPU time is saved in routers and hosts (clients, servers, proxies, gateways, tunnels, or caches), and memory used for TCP protocol control blocks can be saved in hosts.

- HTTP requests and responses can be pipelined on a connection. Pipelining allows a client to make multiple requests without waiting for each response, allowing a single TCP connection to be used much more efficiently, with much lower elapsed time.

- Network congestion is reduced by reducing the number of packets caused by TCP opens, and by allowing TCP sufficient time to determine the congestion state of the network.

- Latency on subsequent requests is reduced since there is no time spent in TCP's connection opening handshake.

- HTTP can evolve more gracefully, since errors can be reported without the penalty of closing the TCP connection. Clients using future versions of HTTP might optimistically try a new feature, but if communicating with an older server, retry with old semantics after an error is reported.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

HTTP implementations SHOULD implement persistent connections.





### 8.1.2 Overall Operation

A significant difference between HTTP/1.1 and earlier versions of HTTP is that persistent connections are the default behavior of any HTTP connection. That is, unless otherwise indicated, the client SHOULD assume that the server will maintain a persistent connection, even after error responses from the server.

Persistent connections provide a mechanism by which a client and a server can signal the close of a TCP connection. This signaling takes place using the Connection header field (section 14.10). Once a close has been signaled, the client MUST NOT send any more requests on that connection.

#### 8.1.2.1 Negotiation

An HTTP/1.1 server MAY assume that a HTTP/1.1 client intends to maintain a persistent connection unless a Connection header including the connection-token "close" was sent in the request. If the server chooses to close the connection immediately after sending the response, it SHOULD send a Connection header including the connection-token close.

An HTTP/1.1 client MAY expect a connection to remain open, but would decide to keep it open based on whether the response from a server contains a Connection header with the connection-token close. In case the client does not want to maintain a connection for more than that request, it SHOULD send a Connection header including the connection-token close.

If either the client or the server sends the close token in the Connection header, that request becomes the last one for the connection.

Clients and servers SHOULD NOT assume that a persistent connection is maintained for HTTP versions less than 1.1 unless it is explicitly signaled. See section 19.6.2 for more information on backward compatibility with HTTP/1.0 clients.

In order to remain persistent, all messages on the connection MUST have a self-defined message length (i.e., one not defined by closure of the connection), as described in section 4.4.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

8.1.2.2 Pipelining

A client that supports persistent connections MAY "pipeline" its requests (i.e., send multiple requests without waiting for each response). A server MUST send its responses to those requests in the same order that the requests were received.

Clients which assume persistent connections and pipeline immediately after connection establishment SHOULD be prepared to retry their connection if the first pipelined attempt fails. If a client does such a retry, it MUST NOT pipeline before it knows the connection is persistent. Clients MUST also be prepared to resend their requests if the server closes the connection before sending all of the corresponding responses.

Clients SHOULD NOT pipeline requests using non-idempotent methods or non-idempotent sequences of methods (see section 9.1.2). Otherwise, a premature termination of the transport connection could lead to indeterminate results. A client wishing to send a non-idempotent request SHOULD wait to send that request until it has received the response status for the previous request.

8.1.3 Proxy Servers

It is especially important that proxies correctly implement the properties of the Connection header field as specified in section 14.10.

The proxy server MUST signal persistent connections separately with its clients and the origin servers (or other proxy servers) that it connects to. Each persistent connection applies to only one transport link.

A proxy server MUST NOT establish a HTTP/1.1 persistent connection with an HTTP/1.0 client (but see RFC 2068 [33] for information and discussion of the problems with the Keep-Alive header implemented by many HTTP/1.0 clients).

8.1.4 Practical Considerations

Servers will usually have some time-out value beyond which they will no longer maintain an inactive connection. Proxy servers might make this a higher value since it is likely that the client will be making more connections through the same server. The use of persistent connections places no requirements on the length (or existence) of this time-out for either the client or the server.



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

When a client or server wishes to time-out it SHOULD issue a graceful close on the transport connection. Clients and servers SHOULD both constantly watch for the other side of the transport close, and respond to it as appropriate. If a client or server does not detect the other side's close promptly it could cause unnecessary resource drain on the network.

A client, server, or proxy MAY close the transport connection at any time. For example, a client might have started to send a new request at the same time that the server has decided to close the "idle" connection. From the server's point of view, the connection is being closed while it was idle, but from the client's point of view, a request is in progress.

This means that clients, servers, and proxies MUST be able to recover from asynchronous close events. Client software SHOULD reopen the transport connection and retransmit the aborted sequence of requests without user interaction so long as the request sequence is idempotent (see section 9.1.2). Non-idempotent methods or sequences MUST NOT be automatically retried, although user agents MAY offer a human operator the choice of retrying the request(s). Confirmation by user-agent software with semantic understanding of the application MAY substitute for user confirmation. The automatic retry SHOULD NOT be repeated if the second sequence of requests fails.

Servers SHOULD always respond to at least one request per connection, if at all possible. Servers SHOULD NOT close a connection in the middle of transmitting a response, unless a network or client failure is suspected.

Clients that use persistent connections SHOULD limit the number of simultaneous connections that they maintain to a given server. A single-user client SHOULD NOT maintain more than 2 connections with any server or proxy. A proxy SHOULD use up to 2*N connections to another server or proxy, where N is the number of simultaneously active users. These guidelines are intended to improve HTTP response times and avoid congestion.

## 8.2 Message Transmission Requirements

### 8.2.1 Persistent Connections and Flow Control

HTTP/1.1 servers SHOULD maintain persistent connections and use TCP's flow control mechanisms to resolve temporary overloads, rather than terminating connections with the expectation that clients will retry. The latter technique can exacerbate network congestion.





### 8.2.2 Monitoring Connections for Error Status Messages

An HTTP/1.1 (or later) client sending a message-body SHOULD monitor the network connection for an error status while it is transmitting the request. If the client sees an error status, it SHOULD immediately cease transmitting the body. If the body is being sent using a "chunked" encoding (section 3.6), a zero length chunk and

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1119
Keysight, EX-1045
Page 1119

empty trailer MAY be used to prematurely mark the end of the message. If the body was preceded by a Content-Length header, the client MUST close the connection.

### 8.2.3 Use of the 100 (Continue) Status

The purpose of the 100 (Continue) status (see section 10.1.1) is to allow a client that is sending a request message with a request body to determine if the origin server is willing to accept the request (based on the request headers) before the client sends the request body. In some cases, it might either be inappropriate or highly inefficient for the client to send the body if the server will reject the message without looking at the body.

Requirements for HTTP/1.1 clients:

- If a client will wait for a 100 (Continue) response before sending the request body, it MUST send an Expect request-header field (section 14.20) with the "100-continue" expectation.

- A client MUST NOT send an Expect request-header field (section 14.20) with the "100-continue" expectation if it does not intend to send a request body.

Because of the presence of older implementations, the protocol allows ambiguous situations in which a client may send "Expect: 100-continue" without receiving either a 417 (Expectation Failed) status or a 100 (Continue) status. Therefore, when a client sends this header field to an origin server (possibly via a proxy) from which it has never seen a 100 (Continue) status, the client SHOULD NOT wait for an indefinite period before sending the request body.

Requirements for HTTP/1.1 origin servers:

- Upon receiving a request which includes an Expect request-header field with the "100-continue" expectation, an origin server MUST either respond with 100 (Continue) status and continue to read from the input stream, or respond with a final status code. The origin server MUST NOT wait for the request body before sending the 100 (Continue) response. If it responds with a final status code, it MAY close the transport connection or it MAY continue





to read and discard the rest of the request. It MUST NOT perform the requested method if it returns a final status code.

- An origin server SHOULD NOT send a 100 (Continue) response if the request message does not include an Expect request-header field with the "100-continue" expectation, and MUST NOT send a 100 (Continue) response if such a request comes from an HTTP/1.0 (or earlier) client. There is an exception to this rule: for compatibility with RFC 2068, a server MAY send a 100 (Continue) status in response to an HTTP/1.1 PUT or POST request that does not include an Expect request-header field with the "100-continue" expectation. This exception, the purpose of which is to minimize any client processing delays associated with an undeclared wait for 100 (Continue) status, applies only to

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

HTTP/1.1 requests, and not to requests with any other HTTP-version value.

- An origin server MAY omit a 100 (Continue) response if it has already received some or all of the request body for the corresponding request.

- An origin server that sends a 100 (Continue) response MUST ultimately send a final status code, once the request body is received and processed, unless it terminates the transport connection prematurely.

- If an origin server receives a request that does not include an Expect request-header field with the "100-continue" expectation, the request includes a request body, and the server responds with a final status code before reading the entire request body from the transport connection, then the server SHOULD NOT close the transport connection until it has read the entire request, or until the client closes the connection. Otherwise, the client might not reliably receive the response message. However, this requirement is not be construed as preventing a server from defending itself against denial-of-service attacks, or from badly broken client implementations.

Requirements for HTTP/1.1 proxies:

- If a proxy receives a request that includes an Expect request-header field with the "100-continue" expectation, and the proxy either knows that the next-hop server complies with HTTP/1.1 or higher, or does not know the HTTP version of the next-hop server, it MUST forward the request, including the Expect header field.





- If the proxy knows that the version of the next-hop server is HTTP/1.0 or lower, it MUST NOT forward the request, and it MUST respond with a 417 (Expectation Failed) status.

- Proxies SHOULD maintain a cache recording the HTTP version numbers received from recently-referenced next-hop servers.

- A proxy MUST NOT forward a 100 (Continue) response if the request message was received from an HTTP/1.0 (or earlier) client and did not include an Expect request-header field with the "100-continue" expectation. This requirement overrides the general rule for forwarding of 1xx responses (see section 10.1).

### 8.2.4 Client Behavior if Server Prematurely Closes Connection

If an HTTP/1.1 client sends a request which includes a request body, but which does not include an Expect request-header field with the "100-continue" expectation, and if the client is not directly connected to an HTTP/1.1 origin server, and if the client sees the connection close before receiving any status from the server, the client SHOULD retry the request. If the client does retry this

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1121
Keysight, EX-1045
Page 1121

request, it MAY use the following "binary exponential backoff" algorithm to be assured of obtaining a reliable response:

1. Initiate a new connection to the server

2. Transmit the request-headers

3. Initialize a variable R to the estimated round-trip time to the server (e.g., based on the time it took to establish the connection), or to a constant value of 5 seconds if the round-trip time is not available.

4. Compute T = R * (2**N), where N is the number of previous retries of this request.

5. Wait either for an error response from the server, or for T seconds (whichever comes first)

6. If no error response is received, after T seconds transmit the body of the request.

7. If client sees that the connection is closed prematurely, repeat from step 1 until the request is accepted, an error response is received, or the user becomes impatient and terminates the retry process.





If at any point an error status is received, the client

- SHOULD NOT continue and

- SHOULD close the connection if it has not completed sending the request message.

## 9 Method Definitions

The set of common methods for HTTP/1.1 is defined below. Although this set can be expanded, additional methods cannot be assumed to share the same semantics for separately extended clients and servers.

The Host request-header field (section 14.23) MUST accompany all HTTP/1.1 requests.

### 9.1 Safe and Idempotent Methods

#### 9.1.1 Safe Methods

Implementors should be aware that the software represents the user in their interactions over the Internet, and should be careful to allow the user to be aware of any actions they might take which may have an unexpected significance to themselves or others.

In particular, the convention has been established that the GET and HEAD methods SHOULD NOT have the significance of taking an action other than retrieval. These methods ought to be considered "safe".

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

This allows user agents to represent other methods, such as POST, PUT and DELETE, in a special way, so that the user is made aware of the fact that a possibly unsafe action is being requested.

Naturally, it is not possible to ensure that the server does not generate side-effects as a result of performing a GET request; in fact, some dynamic resources consider that a feature. The important distinction here is that the user did not request the side-effects, so therefore cannot be held accountable for them.

### 9.1.2 Idempotent Methods

Methods can also have the property of "idempotence" in that (aside from error or expiration issues) the side-effects of N > 0 identical requests is the same as for a single request. The methods GET, HEAD, PUT and DELETE share this property. Also, the methods OPTIONS and TRACE SHOULD NOT have side effects, and so are inherently idempotent.





However, it is possible that a sequence of several requests is non-idempotent, even if all of the methods executed in that sequence are idempotent. (A sequence is idempotent if a single execution of the entire sequence always yields a result that is not changed by a reexecution of all, or part, of that sequence.) For example, a sequence is non-idempotent if its result depends on a value that is later modified in the same sequence.

A sequence that never has side effects is idempotent, by definition (provided that no concurrent operations are being executed on the same set of resources).

## 9.2 OPTIONS

The OPTIONS method represents a request for information about the communication options available on the request/response chain identified by the Request-URI. This method allows the client to determine the options and/or requirements associated with a resource, or the capabilities of a server, without implying a resource action or initiating a resource retrieval.

Responses to this method are not cacheable.

If the OPTIONS request includes an entity-body (as indicated by the presence of Content-Length or Transfer-Encoding), then the media type MUST be indicated by a Content-Type field. Although this specification does not define any use for such a body, future extensions to HTTP might use the OPTIONS body to make more detailed queries on the server. A server that does not support such an extension MAY discard the request body.

If the Request-URI is an asterisk ("*"), the OPTIONS request is intended to apply to the server in general rather than to a specific resource. Since a server's communication options typically depend on the resource, the "*" request is only useful as a "ping" or "no-op"

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

type of method; it does nothing beyond allowing the client to test the capabilities of the server. For example, this can be used to test a proxy for HTTP/1.1 compliance (or lack thereof).

If the Request-URI is not an asterisk, the OPTIONS request applies only to the options that are available when communicating with that resource.

A 200 response SHOULD include any header fields that indicate optional features implemented by the server and applicable to that resource (e.g., Allow), possibly including extensions not defined by this specification. The response body, if any, SHOULD also include information about the communication options. The format for such a





body is not defined by this specification, but might be defined by future extensions to HTTP. Content negotiation MAY be used to select the appropriate response format. If no response body is included, the response MUST include a Content-Length field with a field-value of "0".

The Max-Forwards request-header field MAY be used to target a specific proxy in the request chain. When a proxy receives an OPTIONS request on an absoluteURI for which request forwarding is permitted, the proxy MUST check for a Max-Forwards field. If the Max-Forwards field-value is zero ("0"), the proxy MUST NOT forward the message; instead, the proxy SHOULD respond with its own communication options. If the Max-Forwards field-value is an integer greater than zero, the proxy MUST decrement the field-value when it forwards the request. If no Max-Forwards field is present in the request, then the forwarded request MUST NOT include a Max-Forwards field.

### 9.3 GET

The GET method means retrieve whatever information (in the form of an entity) is identified by the Request-URI. If the Request-URI refers to a data-producing process, it is the produced data which shall be returned as the entity in the response and not the source text of the process, unless that text happens to be the output of the process.

The semantics of the GET method change to a "conditional GET" if the request message includes an If-Modified-Since, If-Unmodified-Since, If-Match, If-None-Match, or If-Range header field. A conditional GET method requests that the entity be transferred only under the circumstances described by the conditional header field(s). The conditional GET method is intended to reduce unnecessary network usage by allowing cached entities to be refreshed without requiring multiple requests or transferring data already held by the client.

The semantics of the GET method change to a "partial GET" if the request message includes a Range header field. A partial GET requests that only part of the entity be transferred, as described in section 14.35. The partial GET method is intended to reduce unnecessary network usage by allowing partially-retrieved entities to be completed without transferring data already held by the client.

The response to a GET request is cacheable if and only if it meets

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

the requirements for HTTP caching described in section 13.

See section 15.1.3 for security considerations when used for forms.





## 9.4 HEAD

The HEAD method is identical to GET except that the server MUST NOT return a message-body in the response. The metainformation contained in the HTTP headers in response to a HEAD request SHOULD be identical to the information sent in response to a GET request. This method can be used for obtaining metainformation about the entity implied by the request without transferring the entity-body itself. This method is often used for testing hypertext links for validity, accessibility, and recent modification.

The response to a HEAD request MAY be cacheable in the sense that the information contained in the response MAY be used to update a previously cached entity from that resource. If the new field values indicate that the cached entity differs from the current entity (as would be indicated by a change in Content-Length, Content-MD5, ETag or Last-Modified), then the cache MUST treat the cache entry as stale.

## 9.5 POST

The POST method is used to request that the origin server accept the entity enclosed in the request as a new subordinate of the resource identified by the Request-URI in the Request-Line. POST is designed to allow a uniform method to cover the following functions:

- Annotation of existing resources;
- Posting a message to a bulletin board, newsgroup, mailing list, or similar group of articles;
- Providing a block of data, such as the result of submitting a form, to a data-handling process;
- Extending a database through an append operation.

The actual function performed by the POST method is determined by the server and is usually dependent on the Request-URI. The posted entity is subordinate to that URI in the same way that a file is subordinate to a directory containing it, a news article is subordinate to a newsgroup to which it is posted, or a record is subordinate to a database.

The action performed by the POST method might not result in a resource that can be identified by a URI. In this case, either 200 (OK) or 204 (No Content) is the appropriate response status, depending on whether or not the response includes an entity that describes the result.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

If a resource has been created on the origin server, the response SHOULD be 201 (Created) and contain an entity which describes the status of the request and refers to the new resource, and a Location header (see section 14.30).

Responses to this method are not cacheable, unless the response includes appropriate Cache-Control or Expires header fields. However, the 303 (See Other) response can be used to direct the user agent to retrieve a cacheable resource.

POST requests MUST obey the message transmission requirements set out in section 8.2.

See section 15.1.3 for security considerations.

### 9.6 PUT

The PUT method requests that the enclosed entity be stored under the supplied Request-URI. If the Request-URI refers to an already existing resource, the enclosed entity SHOULD be considered as a modified version of the one residing on the origin server. If the Request-URI does not point to an existing resource, and that URI is capable of being defined as a new resource by the requesting user agent, the origin server can create the resource with that URI. If a new resource is created, the origin server MUST inform the user agent via the 201 (Created) response. If an existing resource is modified, either the 200 (OK) or 204 (No Content) response codes SHOULD be sent to indicate successful completion of the request. If the resource could not be created or modified with the Request-URI, an appropriate error response SHOULD be given that reflects the nature of the problem. The recipient of the entity MUST NOT ignore any Content-* (e.g. Content-Range) headers that it does not understand or implement and MUST return a 501 (Not Implemented) response in such cases.

If the request passes through a cache and the Request-URI identifies one or more currently cached entities, those entries SHOULD be treated as stale. Responses to this method are not cacheable.

The fundamental difference between the POST and PUT requests is reflected in the different meaning of the Request-URI. The URI in a POST request identifies the resource that will handle the enclosed entity. That resource might be a data-accepting process, a gateway to some other protocol, or a separate entity that accepts annotations. In contrast, the URI in a PUT request identifies the entity enclosed with the request -- the user agent knows what URI is intended and the server MUST NOT attempt to apply the request to some other resource. If the server desires that the request be applied to a different URI,



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

it MUST send a 301 (Moved Permanently) response; the user agent MAY then make its own decision regarding whether or not to redirect the request.

A single resource MAY be identified by many different URIs. For example, an article might have a URI for identifying "the current version" which is separate from the URI identifying each particular version. In this case, a PUT request on a general URI might result in several other URIs being defined by the origin server.

HTTP/1.1 does not define how a PUT method affects the state of an origin server.

PUT requests MUST obey the message transmission requirements set out in section 8.2.

Unless otherwise specified for a particular entity-header, the entity-headers in the PUT request SHOULD be applied to the resource created or modified by the PUT.

### 9.7 DELETE

The DELETE method requests that the origin server delete the resource identified by the Request-URI. This method MAY be overridden by human intervention (or other means) on the origin server. The client cannot be guaranteed that the operation has been carried out, even if the status code returned from the origin server indicates that the action has been completed successfully. However, the server SHOULD NOT indicate success unless, at the time the response is given, it intends to delete the resource or move it to an inaccessible location.

A successful response SHOULD be 200 (OK) if the response includes an entity describing the status, 202 (Accepted) if the action has not yet been enacted, or 204 (No Content) if the action has been enacted but the response does not include an entity.

If the request passes through a cache and the Request-URI identifies one or more currently cached entities, those entries SHOULD be treated as stale. Responses to this method are not cacheable.

### 9.8 TRACE

The TRACE method is used to invoke a remote, application-layer loop-back of the request message. The final recipient of the request SHOULD reflect the message received back to the client as the entity-body of a 200 (OK) response. The final recipient is either the





origin server or the first proxy or gateway to receive a Max-Forwards value of zero (0) in the request (see section 14.31). A TRACE request MUST NOT include an entity.

TRACE allows the client to see what is being received at the other end of the request chain and use that data for testing or diagnostic information. The value of the Via header field (section 14.45) is of

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1127
Keysight, EX-1045
Page 1127

particular interest, since it acts as a trace of the request chain. Use of the Max-Forwards header field allows the client to limit the length of the request chain, which is useful for testing a chain of proxies forwarding messages in an infinite loop.

If the request is valid, the response SHOULD contain the entire request message in the entity-body, with a Content-Type of "message/http". Responses to this method MUST NOT be cached.

### 9.9 CONNECT

This specification reserves the method name CONNECT for use with a proxy that can dynamically switch to being a tunnel (e.g. SSL tunneling [44]).

## 10 Status Code Definitions

Each Status-Code is described below, including a description of which method(s) it can follow and any metainformation required in the response.

### 10.1 Informational 1xx

This class of status code indicates a provisional response, consisting only of the Status-Line and optional headers, and is terminated by an empty line. There are no required headers for this class of status code. Since HTTP/1.0 did not define any 1xx status codes, servers MUST NOT send a 1xx response to an HTTP/1.0 client except under experimental conditions.

A client MUST be prepared to accept one or more 1xx status responses prior to a regular response, even if the client does not expect a 100 (Continue) status message. Unexpected 1xx status responses MAY be ignored by a user agent.

Proxies MUST forward 1xx responses, unless the connection between the proxy and its client has been closed, or unless the proxy itself requested the generation of the 1xx response. (For example, if a





proxy adds a "Expect: 100-continue" field when it forwards a request, then it need not forward the corresponding 100 (Continue) response(s).)

#### 10.1.1 100 Continue

The client SHOULD continue with its request. This interim response is used to inform the client that the initial part of the request has been received and has not yet been rejected by the server. The client SHOULD continue by sending the remainder of the request or, if the request has already been completed, ignore this response. The server MUST send a final response after the request has been completed. See section 8.2.3 for detailed discussion of the use and handling of this status code.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

10.1.2 101 Switching Protocols

The server understands and is willing to comply with the client's request, via the Upgrade message header field (section 14.42), for a change in the application protocol being used on this connection. The server will switch protocols to those defined by the response's Upgrade header field immediately after the empty line which terminates the 101 response.

The protocol SHOULD be switched only when it is advantageous to do so. For example, switching to a newer version of HTTP is advantageous over older versions, and switching to a real-time, synchronous protocol might be advantageous when delivering resources that use such features.

10.2 Successful 2xx

This class of status code indicates that the client's request was successfully received, understood, and accepted.

10.2.1 200 OK

The request has succeeded. The information returned with the response is dependent on the method used in the request, for example:

GET an entity corresponding to the requested resource is sent in the response;

HEAD the entity-header fields corresponding to the requested resource are sent in the response without any message-body;

POST an entity describing or containing the result of the action;





TRACE an entity containing the request message as received by the end server.

10.2.2 201 Created

The request has been fulfilled and resulted in a new resource being created. The newly created resource can be referenced by the URI(s) returned in the entity of the response, with the most specific URI for the resource given by a Location header field. The response SHOULD include an entity containing a list of resource characteristics and location(s) from which the user or user agent can choose the one most appropriate. The entity format is specified by the media type given in the Content-Type header field. The origin server MUST create the resource before returning the 201 status code. If the action cannot be carried out immediately, the server SHOULD respond with 202 (Accepted) response instead.

A 201 response MAY contain an ETag response header field indicating the current value of the entity tag for the requested variant just created, see section 14.19.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

10.2.3 202 Accepted

The request has been accepted for processing, but the processing has not been completed. The request might or might not eventually be acted upon, as it might be disallowed when processing actually takes place. There is no facility for re-sending a status code from an asynchronous operation such as this.

The 202 response is intentionally non-committal. Its purpose is to allow a server to accept a request for some other process (perhaps a batch-oriented process that is only run once per day) without requiring that the user agent's connection to the server persist until the process is completed. The entity returned with this response SHOULD include an indication of the request's current status and either a pointer to a status monitor or some estimate of when the user can expect the request to be fulfilled.

10.2.4 203 Non-Authoritative Information

The returned metainformation in the entity-header is not the definitive set as available from the origin server, but is gathered from a local or a third-party copy. The set presented MAY be a subset or superset of the original version. For example, including local annotation information about the resource might result in a superset of the metainformation known by the origin server. Use of this response code is not required and is only appropriate when the response would otherwise be 200 (OK).





10.2.5 204 No Content

The server has fulfilled the request but does not need to return an entity-body, and might want to return updated metainformation. The response MAY include new or updated metainformation in the form of entity-headers, which if present SHOULD be associated with the requested variant.

If the client is a user agent, it SHOULD NOT change its document view from that which caused the request to be sent. This response is primarily intended to allow input for actions to take place without causing a change to the user agent's active document view, although any new or updated metainformation SHOULD be applied to the document currently in the user agent's active view.

The 204 response MUST NOT include a message-body, and thus is always terminated by the first empty line after the header fields.

10.2.6 205 Reset Content

The server has fulfilled the request and the user agent SHOULD reset the document view which caused the request to be sent. This response is primarily intended to allow input for actions to take place via user input, followed by a clearing of the form in which the input is given so that the user can easily initiate another input action. The response MUST NOT include an entity.

10.2.7 206 Partial Content

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The server has fulfilled the partial GET request for the resource. The request MUST have included a Range header field (section 14.35) indicating the desired range, and MAY have included an If-Range header field (section 14.27) to make the request conditional.

The response MUST include the following header fields:

- Either a Content-Range header field (section 14.16) indicating the range included with this response, or a multipart/byteranges Content-Type including Content-Range fields for each part. If a Content-Length header field is present in the response, its value MUST match the actual number of OCTETs transmitted in the message-body.

- Date

- ETag and/or Content-Location, if the header would have been sent in a 200 response to the same request





- Expires, Cache-Control, and/or Vary, if the field-value might differ from that sent in any previous response for the same variant

If the 206 response is the result of an If-Range request that used a strong cache validator (see section 13.3.3), the response SHOULD NOT include other entity-headers. If the response is the result of an If-Range request that used a weak validator, the response MUST NOT include other entity-headers; this prevents inconsistencies between cached entity-bodies and updated headers. Otherwise, the response MUST include all of the entity-headers that would have been returned with a 200 (OK) response to the same request.

A cache MUST NOT combine a 206 response with other previously cached content if the ETag or Last-Modified headers do not match exactly, see 13.5.4.

A cache that does not support the Range and Content-Range headers MUST NOT cache 206 (Partial) responses.

### 10.3 Redirection 3xx

This class of status code indicates that further action needs to be taken by the user agent in order to fulfill the request. The action required MAY be carried out by the user agent without interaction with the user if and only if the method used in the second request is GET or HEAD. A client SHOULD detect infinite redirection loops, since such loops generate network traffic for each redirection.

Note: previous versions of this specification recommended a maximum of five redirections. Content developers should be aware that there might be clients that implement such a fixed limitation.

#### 10.3.1 300 Multiple Choices

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The requested resource corresponds to any one of a set of representations, each with its own specific location, and agent-driven negotiation information (section 12) is being provided so that the user (or user agent) can select a preferred representation and redirect its request to that location.

Unless it was a HEAD request, the response SHOULD include an entity containing a list of resource characteristics and location(s) from which the user or user agent can choose the one most appropriate. The entity format is specified by the media type given in the Content-Type header field. Depending upon the format and the capabilities of





the user agent, selection of the most appropriate choice MAY be performed automatically. However, this specification does not define any standard for such automatic selection.

If the server has a preferred choice of representation, it SHOULD include the specific URI for that representation in the Location field; user agents MAY use the Location field value for automatic redirection. This response is cacheable unless indicated otherwise.

### 10.3.2 301 Moved Permanently

The requested resource has been assigned a new permanent URI and any future references to this resource SHOULD use one of the returned URIs. Clients with link editing capabilities ought to automatically re-link references to the Request-URI to one or more of the new references returned by the server, where possible. This response is cacheable unless indicated otherwise.

The new permanent URI SHOULD be given by the Location field in the response. Unless the request method was HEAD, the entity of the response SHOULD contain a short hypertext note with a hyperlink to the new URI(s).

If the 301 status code is received in response to a request other than GET or HEAD, the user agent MUST NOT automatically redirect the request unless it can be confirmed by the user, since this might change the conditions under which the request was issued.

> Note: When automatically redirecting a POST request after receiving a 301 status code, some existing HTTP/1.0 user agents will erroneously change it into a GET request.

### 10.3.3 302 Found

The requested resource resides temporarily under a different URI. Since the redirection might be altered on occasion, the client SHOULD continue to use the Request-URI for future requests. This response is only cacheable if indicated by a Cache-Control or Expires header field.

The temporary URI SHOULD be given by the Location field in the response. Unless the request method was HEAD, the entity of the

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

response SHOULD contain a short hypertext note with a hyperlink to the new URI(s).





If the 302 status code is received in response to a request other than GET or HEAD, the user agent MUST NOT automatically redirect the request unless it can be confirmed by the user, since this might change the conditions under which the request was issued.

Note: RFC 1945 and RFC 2068 specify that the client is not allowed to change the method on the redirected request. However, most existing user agent implementations treat 302 as if it were a 303 response, performing a GET on the Location field-value regardless of the original request method. The status codes 303 and 307 have been added for servers that wish to make unambiguously clear which kind of reaction is expected of the client.

### 10.3.4 303 See Other

The response to the request can be found under a different URI and SHOULD be retrieved using a GET method on that resource. This method exists primarily to allow the output of a POST-activated script to redirect the user agent to a selected resource. The new URI is not a substitute reference for the originally requested resource. The 303 response MUST NOT be cached, but the response to the second (redirected) request might be cacheable.

The different URI SHOULD be given by the Location field in the response. Unless the request method was HEAD, the entity of the response SHOULD contain a short hypertext note with a hyperlink to the new URI(s).

Note: Many pre-HTTP/1.1 user agents do not understand the 303 status. When interoperability with such clients is a concern, the 302 status code may be used instead, since most user agents react to a 302 response as described here for 303.

### 10.3.5 304 Not Modified

If the client has performed a conditional GET request and access is allowed, but the document has not been modified, the server SHOULD respond with this status code. The 304 response MUST NOT contain a message-body, and thus is always terminated by the first empty line after the header fields.

The response MUST include the following header fields:

- Date, unless its omission is required by section 14.18.1

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

If a clockless origin server obeys these rules, and proxies and clients add their own Date to any response received without one (as already specified by [RFC 2068], section 14.19), caches will operate correctly.

- ETag and/or Content-Location, if the header would have been sent in a 200 response to the same request

- Expires, Cache-Control, and/or Vary, if the field-value might differ from that sent in any previous response for the same variant

If the conditional GET used a strong cache validator (see section 13.3.3), the response SHOULD NOT include other entity-headers. Otherwise (i.e., the conditional GET used a weak validator), the response MUST NOT include other entity-headers; this prevents inconsistencies between cached entity-bodies and updated headers.

If a 304 response indicates an entity not currently cached, then the cache MUST disregard the response and repeat the request without the conditional.

If a cache uses a received 304 response to update a cache entry, the cache MUST update the entry to reflect any new field values given in the response.

### 10.3.6 305 Use Proxy

The requested resource MUST be accessed through the proxy given by the Location field. The Location field gives the URI of the proxy. The recipient is expected to repeat this single request via the proxy. 305 responses MUST only be generated by origin servers.

> Note: RFC 2068 was not clear that 305 was intended to redirect a single request, and to be generated by origin servers only. Not observing these limitations has significant security consequences.

### 10.3.7 306 (Unused)

The 306 status code was used in a previous version of the specification, is no longer used, and the code is reserved.



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

### 10.3.8 307 Temporary Redirect

The requested resource resides temporarily under a different URI. Since the redirection MAY be altered on occasion, the client SHOULD continue to use the Request-URI for future requests. This response is only cacheable if indicated by a Cache-Control or Expires header field.

The temporary URI SHOULD be given by the Location field in the response. Unless the request method was HEAD, the entity of the response SHOULD contain a short hypertext note with a hyperlink to the new URI(s) , since many pre-HTTP/1.1 user agents do not understand the 307 status. Therefore, the note SHOULD contain the information necessary for a user to repeat the original request on the new URI.

If the 307 status code is received in response to a request other than GET or HEAD, the user agent MUST NOT automatically redirect the request unless it can be confirmed by the user, since this might change the conditions under which the request was issued.

## 10.4 Client Error 4xx

The 4xx class of status code is intended for cases in which the client seems to have erred. Except when responding to a HEAD request, the server SHOULD include an entity containing an explanation of the error situation, and whether it is a temporary or permanent condition. These status codes are applicable to any request method. User agents SHOULD display any included entity to the user.

If the client is sending data, a server implementation using TCP SHOULD be careful to ensure that the client acknowledges receipt of the packet(s) containing the response, before the server closes the input connection. If the client continues sending data to the server after the close, the server's TCP stack will send a reset packet to the client, which may erase the client's unacknowledged input buffers before they can be read and interpreted by the HTTP application.

### 10.4.1 400 Bad Request

The request could not be understood by the server due to malformed syntax. The client SHOULD NOT repeat the request without modifications.





### 10.4.2 401 Unauthorized

The request requires user authentication. The response MUST include a WWW-Authenticate header field (section 14.47) containing a challenge applicable to the requested resource. The client MAY repeat the request with a suitable Authorization header field (section 14.8). If the request already included Authorization credentials, then the 401

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1135
Keysight, EX-1045
Page 1135

response indicates that authorization has been refused for those credentials. If the 401 response contains the same challenge as the prior response, and the user agent has already attempted authentication at least once, then the user SHOULD be presented the entity that was given in the response, since that entity might include relevant diagnostic information. HTTP access authentication is explained in "HTTP Authentication: Basic and Digest Access Authentication" [43].

10.4.3 402 Payment Required

This code is reserved for future use.

10.4.4 403 Forbidden

The server understood the request, but is refusing to fulfill it. Authorization will not help and the request SHOULD NOT be repeated. If the request method was not HEAD and the server wishes to make public why the request has not been fulfilled, it SHOULD describe the reason for the refusal in the entity. If the server does not wish to make this information available to the client, the status code 404 (Not Found) can be used instead.

10.4.5 404 Not Found

The server has not found anything matching the Request-URI. No indication is given of whether the condition is temporary or permanent. The 410 (Gone) status code SHOULD be used if the server knows, through some internally configurable mechanism, that an old resource is permanently unavailable and has no forwarding address. This status code is commonly used when the server does not wish to reveal exactly why the request has been refused, or when no other response is applicable.

10.4.6 405 Method Not Allowed

The method specified in the Request-Line is not allowed for the resource identified by the Request-URI. The response MUST include an Allow header containing a list of valid methods for the requested resource.





10.4.7 406 Not Acceptable

The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request.

Unless it was a HEAD request, the response SHOULD include an entity containing a list of available entity characteristics and location(s) from which the user or user agent can choose the one most appropriate. The entity format is specified by the media type given in the Content-Type header field. Depending upon the format and the capabilities of the user agent, selection of the most appropriate choice MAY be performed automatically. However, this specification does not define any standard for such automatic selection.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

Note: HTTP/1.1 servers are allowed to return responses which are not acceptable according to the accept headers sent in the request. In some cases, this may even be preferable to sending a 406 response. User agents are encouraged to inspect the headers of an incoming response to determine if it is acceptable.

If the response could be unacceptable, a user agent SHOULD temporarily stop receipt of more data and query the user for a decision on further actions.

10.4.8 407 Proxy Authentication Required

This code is similar to 401 (Unauthorized), but indicates that the client must first authenticate itself with the proxy. The proxy MUST return a Proxy-Authenticate header field (section 14.33) containing a challenge applicable to the proxy for the requested resource. The client MAY repeat the request with a suitable Proxy-Authorization header field (section 14.34). HTTP access authentication is explained in "HTTP Authentication: Basic and Digest Access Authentication" [43].

10.4.9 408 Request Timeout

The client did not produce a request within the time that the server was prepared to wait. The client MAY repeat the request without modifications at any later time.

10.4.10 409 Conflict

The request could not be completed due to a conflict with the current state of the resource. This code is only allowed in situations where it is expected that the user might be able to resolve the conflict and resubmit the request. The response body SHOULD include enough





information for the user to recognize the source of the conflict. Ideally, the response entity would include enough information for the user or user agent to fix the problem; however, that might not be possible and is not required.

Conflicts are most likely to occur in response to a PUT request. For example, if versioning were being used and the entity being PUT included changes to a resource which conflict with those made by an earlier (third-party) request, the server might use the 409 response to indicate that it can't complete the request. In this case, the response entity would likely contain a list of the differences between the two versions in a format defined by the response Content-Type.

10.4.11 410 Gone

The requested resource is no longer available at the server and no forwarding address is known. This condition is expected to be considered permanent. Clients with link editing capabilities SHOULD delete references to the Request-URI after user approval. If the server does not know, or has no facility to determine, whether or not

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

the condition is permanent, the status code 404 (Not Found) SHOULD be used instead. This response is cacheable unless indicated otherwise.

The 410 response is primarily intended to assist the task of web maintenance by notifying the recipient that the resource is intentionally unavailable and that the server owners desire that remote links to that resource be removed. Such an event is common for limited-time, promotional services and for resources belonging to individuals no longer working at the server's site. It is not necessary to mark all permanently unavailable resources as "gone" or to keep the mark for any length of time -- that is left to the discretion of the server owner.

10.4.12 411 Length Required

The server refuses to accept the request without a defined Content-Length. The client MAY repeat the request if it adds a valid Content-Length header field containing the length of the message-body in the request message.

10.4.13 412 Precondition Failed

The precondition given in one or more of the request-header fields evaluated to false when it was tested on the server. This response code allows the client to place preconditions on the current resource metainformation (header field data) and thus prevent the requested method from being applied to a resource other than the one intended.





10.4.14 413 Request Entity Too Large

The server is refusing to process a request because the request entity is larger than the server is willing or able to process. The server MAY close the connection to prevent the client from continuing the request.

If the condition is temporary, the server SHOULD include a Retry-After header field to indicate that it is temporary and after what time the client MAY try again.

10.4.15 414 Request-URI Too Long

The server is refusing to service the request because the Request-URI is longer than the server is willing to interpret. This rare condition is only likely to occur when a client has improperly converted a POST request to a GET request with long query information, when the client has descended into a URI "black hole" of redirection (e.g., a redirected URI prefix that points to a suffix of itself), or when the server is under attack by a client attempting to exploit security holes present in some servers using fixed-length buffers for reading or manipulating the Request-URI.

10.4.16 415 Unsupported Media Type

The server is refusing to service the request because the entity of the request is in a format not supported by the requested resource for the requested method.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

10.4.17 416 Requested Range Not Satisfiable

A server SHOULD return a response with this status code if a request included a Range request-header field (section 14.35), and none of the range-specifier values in this field overlap the current extent of the selected resource, and the request did not include an If-Range request-header field. (For byte-ranges, this means that the first-byte-pos of all of the byte-range-spec values were greater than the current length of the selected resource.)

When this status code is returned for a byte-range request, the response SHOULD include a Content-Range entity-header field specifying the current length of the selected resource (see section 14.16). This response MUST NOT use the multipart/byteranges content-type.





10.4.18 417 Expectation Failed

The expectation given in an Expect request-header field (see section 14.20) could not be met by this server, or, if the server is a proxy, the server has unambiguous evidence that the request could not be met by the next-hop server.

10.5 Server Error 5xx

Response status codes beginning with the digit "5" indicate cases in which the server is aware that it has erred or is incapable of performing the request. Except when responding to a HEAD request, the server SHOULD include an entity containing an explanation of the error situation, and whether it is a temporary or permanent condition. User agents SHOULD display any included entity to the user. These response codes are applicable to any request method.

10.5.1 500 Internal Server Error

The server encountered an unexpected condition which prevented it from fulfilling the request.

10.5.2 501 Not Implemented

The server does not support the functionality required to fulfill the request. This is the appropriate response when the server does not recognize the request method and is not capable of supporting it for any resource.

10.5.3 502 Bad Gateway

The server, while acting as a gateway or proxy, received an invalid response from the upstream server it accessed in attempting to fulfill the request.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

10.5.4 503 Service Unavailable

The server is currently unable to handle the request due to a temporary overloading or maintenance of the server. The implication is that this is a temporary condition which will be alleviated after some delay. If known, the length of the delay MAY be indicated in a Retry-After header. If no Retry-After is given, the client SHOULD handle the response as it would for a 500 response.

Note: The existence of the 503 status code does not imply that a server must use it when becoming overloaded. Some servers may wish to simply refuse the connection.





10.5.5 504 Gateway Timeout

The server, while acting as a gateway or proxy, did not receive a timely response from the upstream server specified by the URI (e.g. HTTP, FTP, LDAP) or some other auxiliary server (e.g. DNS) it needed to access in attempting to complete the request.

Note: Note to implementors: some deployed proxies are known to return 400 or 500 when DNS lookups time out.

10.5.6 505 HTTP Version Not Supported

The server does not support, or refuses to support, the HTTP protocol version that was used in the request message. The server is indicating that it is unable or unwilling to complete the request using the same major version as the client, as described in section 3.1, other than with this error message. The response SHOULD contain an entity describing why that version is not supported and what other protocols are supported by that server.

11 Access Authentication

HTTP provides several OPTIONAL challenge-response authentication mechanisms which can be used by a server to challenge a client request and by a client to provide authentication information. The general framework for access authentication, and the specification of "basic" and "digest" authentication, are specified in "HTTP Authentication: Basic and Digest Access Authentication" [43]. This specification adopts the definitions of "challenge" and "credentials" from that specification.

12 Content Negotiation

Most HTTP responses include an entity which contains information for interpretation by a human user. Naturally, it is desirable to supply the user with the "best available" entity corresponding to the request. Unfortunately for servers and caches, not all users have the same preferences for what is "best," and not all user agents are equally capable of rendering all entity types. For that reason, HTTP has provisions for several mechanisms for "content negotiation" -- the process of selecting the best representation for a given response when there are multiple representations available.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

Note: This is not called "format negotiation" because the alternate representations may be of the same media type, but use different capabilities of that type, be in different languages, etc.





Any response containing an entity-body MAY be subject to negotiation, including error responses.

There are two kinds of content negotiation which are possible in HTTP: server-driven and agent-driven negotiation. These two kinds of negotiation are orthogonal and thus may be used separately or in combination. One method of combination, referred to as transparent negotiation, occurs when a cache uses the agent-driven negotiation information provided by the origin server in order to provide server-driven negotiation for subsequent requests.

### 12.1 Server-driven Negotiation

If the selection of the best representation for a response is made by an algorithm located at the server, it is called server-driven negotiation. Selection is based on the available representations of the response (the dimensions over which it can vary; e.g. language, content-coding, etc.) and the contents of particular header fields in the request message or on other information pertaining to the request (such as the network address of the client).

Server-driven negotiation is advantageous when the algorithm for selecting from among the available representations is difficult to describe to the user agent, or when the server desires to send its "best guess" to the client along with the first response (hoping to avoid the round-trip delay of a subsequent request if the "best guess" is good enough for the user). In order to improve the server's guess, the user agent MAY include request header fields (Accept, Accept-Language, Accept-Encoding, etc.) which describe its preferences for such a response.

Server-driven negotiation has disadvantages:

1. It is impossible for the server to accurately determine what might be "best" for any given user, since that would require complete knowledge of both the capabilities of the user agent and the intended use for the response (e.g., does the user want to view it on screen or print it on paper?).

2. Having the user agent describe its capabilities in every request can be both very inefficient (given that only a small percentage of responses have multiple representations) and a potential violation of the user's privacy.

3. It complicates the implementation of an origin server and the algorithms for generating responses to a request.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

4. It may limit a public cache's ability to use the same response for multiple user's requests.

HTTP/1.1 includes the following request-header fields for enabling server-driven negotiation through description of user agent capabilities and user preferences: Accept (section 14.1), Accept-Charset (section 14.2), Accept-Encoding (section 14.3), Accept-Language (section 14.4), and User-Agent (section 14.43). However, an origin server is not limited to these dimensions and MAY vary the response based on any aspect of the request, including information outside the request-header fields or within extension header fields not defined by this specification.

The Vary header field can be used to express the parameters the server uses to select a representation that is subject to server-driven negotiation. See section 13.6 for use of the Vary header field by caches and section 14.44 for use of the Vary header field by servers.

### 12.2 Agent-driven Negotiation

With agent-driven negotiation, selection of the best representation for a response is performed by the user agent after receiving an initial response from the origin server. Selection is based on a list of the available representations of the response included within the header fields or entity-body of the initial response, with each representation identified by its own URI. Selection from among the representations may be performed automatically (if the user agent is capable of doing so) or manually by the user selecting from a generated (possibly hypertext) menu.

Agent-driven negotiation is advantageous when the response would vary over commonly-used dimensions (such as type, language, or encoding), when the origin server is unable to determine a user agent's capabilities from examining the request, and generally when public caches are used to distribute server load and reduce network usage.

Agent-driven negotiation suffers from the disadvantage of needing a second request to obtain the best alternate representation. This second request is only efficient when caching is used. In addition, this specification does not define any mechanism for supporting automatic selection, though it also does not prevent any such mechanism from being developed as an extension and used within HTTP/1.1.



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

HTTP/1.1 defines the 300 (Multiple Choices) and 406 (Not Acceptable) status codes for enabling agent-driven negotiation when the server is unwilling or unable to provide a varying response using server-driven negotiation.

### 12.3 Transparent Negotiation

Transparent negotiation is a combination of both server-driven and agent-driven negotiation. When a cache is supplied with a form of the list of available representations of the response (as in agent-driven negotiation) and the dimensions of variance are completely understood by the cache, then the cache becomes capable of performing server-driven negotiation on behalf of the origin server for subsequent requests on that resource.

Transparent negotiation has the advantage of distributing the negotiation work that would otherwise be required of the origin server and also removing the second request delay of agent-driven negotiation when the cache is able to correctly guess the right response.

This specification does not define any mechanism for transparent negotiation, though it also does not prevent any such mechanism from being developed as an extension that could be used within HTTP/1.1.

## 13 Caching in HTTP

HTTP is typically used for distributed information systems, where performance can be improved by the use of response caches. The HTTP/1.1 protocol includes a number of elements intended to make caching work as well as possible. Because these elements are inextricable from other aspects of the protocol, and because they interact with each other, it is useful to describe the basic caching design of HTTP separately from the detailed descriptions of methods, headers, response codes, etc.

Caching would be useless if it did not significantly improve performance. The goal of caching in HTTP/1.1 is to eliminate the need to send requests in many cases, and to eliminate the need to send full responses in many other cases. The former reduces the number of network round-trips required for many operations; we use an "expiration" mechanism for this purpose (see section 13.2). The latter reduces network bandwidth requirements; we use a "validation" mechanism for this purpose (see section 13.3).

Requirements for performance, availability, and disconnected operation require us to be able to relax the goal of semantic transparency. The HTTP/1.1 protocol allows origin servers, caches,





and clients to explicitly reduce transparency when necessary. However, because non-transparent operation may confuse non-expert users, and might be incompatible with certain server applications (such as those for ordering merchandise), the protocol requires that transparency be relaxed

- only by an explicit protocol-level request when relaxed by

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1143
Keysight, EX-1045
Page 1143

client or origin server

- only with an explicit warning to the end user when relaxed by cache or client

Therefore, the HTTP/1.1 protocol provides these important elements:

1. Protocol features that provide full semantic transparency when this is required by all parties.

2. Protocol features that allow an origin server or user agent to explicitly request and control non-transparent operation.

3. Protocol features that allow a cache to attach warnings to responses that do not preserve the requested approximation of semantic transparency.

A basic principle is that it must be possible for the clients to detect any potential relaxation of semantic transparency.

Note: The server, cache, or client implementor might be faced with design decisions not explicitly discussed in this specification. If a decision might affect semantic transparency, the implementor ought to err on the side of maintaining transparency unless a careful and complete analysis shows significant benefits in breaking transparency.

### 13.1.1 Cache Correctness

A correct cache MUST respond to a request with the most up-to-date response held by the cache that is appropriate to the request (see sections 13.2.5, 13.2.6, and 13.12) which meets one of the following conditions:

1. It has been checked for equivalence with what the origin server would have returned by revalidating the response with the origin server (section 13.3);





2. It is "fresh enough" (see section 13.2). In the default case, this means it meets the least restrictive freshness requirement of the client, origin server, and cache (see section 14.9); if the origin server so specifies, it is the freshness requirement of the origin server alone.

   If a stored response is not "fresh enough" by the most restrictive freshness requirement of both the client and the origin server, in carefully considered circumstances the cache MAY still return the response with the appropriate Warning header (see section 13.1.5 and 14.46), unless such a response is prohibited (e.g., by a "no-store" cache-directive, or by a "no-cache" cache-request-directive; see section 14.9).

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

3. It is an appropriate 304 (Not Modified), 305 (Proxy Redirect), or error (4xx or 5xx) response message.

If the cache can not communicate with the origin server, then a correct cache SHOULD respond as above if the response can be correctly served from the cache; if not it MUST return an error or warning indicating that there was a communication failure.

If a cache receives a response (either an entire response, or a 304 (Not Modified) response) that it would normally forward to the requesting client, and the received response is no longer fresh, the cache SHOULD forward it to the requesting client without adding a new Warning (but without removing any existing Warning headers). A cache SHOULD NOT attempt to revalidate a response simply because that response became stale in transit; this might lead to an infinite loop. A user agent that receives a stale response without a Warning MAY display a warning indication to the user.

13.1.2 Warnings

Whenever a cache returns a response that is neither first-hand nor "fresh enough" (in the sense of condition 2 in section 13.1.1), it MUST attach a warning to that effect, using a Warning general-header. The Warning header and the currently defined warnings are described in section 14.46. The warning allows clients to take appropriate action.

Warnings MAY be used for other purposes, both cache-related and otherwise. The use of a warning, rather than an error status code, distinguish these responses from true failures.

Warnings are assigned three digit warn-codes. The first digit indicates whether the Warning MUST or MUST NOT be deleted from a stored cache entry after a successful revalidation:





1xx Warnings that describe the freshness or revalidation status of the response, and so MUST be deleted after a successful revalidation. 1XX warn-codes MAY be generated by a cache only when validating a cached entry. It MUST NOT be generated by clients.

2xx Warnings that describe some aspect of the entity body or entity headers that is not rectified by a revalidation (for example, a lossy compression of the entity bodies) and which MUST NOT be deleted after a successful revalidation.

See section 14.46 for the definitions of the codes themselves.

HTTP/1.0 caches will cache all Warnings in responses, without deleting the ones in the first category. Warnings in responses that are passed to HTTP/1.0 caches carry an extra warning-date field, which prevents a future HTTP/1.1 recipient from believing an erroneously cached Warning.

Warnings also carry a warning text. The text MAY be in any appropriate natural language (perhaps based on the client's Accept headers), and include an OPTIONAL indication of what character set is

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

used.

Multiple warnings MAY be attached to a response (either by the origin server or by a cache), including multiple warnings with the same code number. For example, a server might provide the same warning with texts in both English and Basque.

When multiple warnings are attached to a response, it might not be practical or reasonable to display all of them to the user. This version of HTTP does not specify strict priority rules for deciding which warnings to display and in what order, but does suggest some heuristics.

### 13.1.3 Cache-control Mechanisms

The basic cache mechanisms in HTTP/1.1 (server-specified expiration times and validators) are implicit directives to caches. In some cases, a server or client might need to provide explicit directives to the HTTP caches. We use the Cache-Control header for this purpose.

The Cache-Control header allows a client or server to transmit a variety of directives in either requests or responses. These directives typically override the default caching algorithms. As a general rule, if there is any apparent conflict between header values, the most restrictive interpretation is applied (that is, the one that is most likely to preserve semantic transparency). However,





in some cases, cache-control directives are explicitly specified as weakening the approximation of semantic transparency (for example, "max-stale" or "public").

The cache-control directives are described in detail in section 14.9.

### 13.1.4 Explicit User Agent Warnings

Many user agents make it possible for users to override the basic caching mechanisms. For example, the user agent might allow the user to specify that cached entities (even explicitly stale ones) are never validated. Or the user agent might habitually add "Cache-Control: max-stale=3600" to every request. The user agent SHOULD NOT default to either non-transparent behavior, or behavior that results in abnormally ineffective caching, but MAY be explicitly configured to do so by an explicit action of the user.

If the user has overridden the basic caching mechanisms, the user agent SHOULD explicitly indicate to the user whenever this results in the display of information that might not meet the server's transparency requirements (in particular, if the displayed entity is known to be stale). Since the protocol normally allows the user agent to determine if responses are stale or not, this indication need only be displayed when this actually happens. The indication need not be a dialog box; it could be an icon (for example, a picture of a rotting fish) or some other indicator.

If the user has overridden the caching mechanisms in a way that would

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

abnormally reduce the effectiveness of caches, the user agent SHOULD continually indicate this state to the user (for example, by a display of a picture of currency in flames) so that the user does not inadvertently consume excess resources or suffer from excessive latency.

### 13.1.5 Exceptions to the Rules and Warnings

In some cases, the operator of a cache MAY choose to configure it to return stale responses even when not requested by clients. This decision ought not be made lightly, but may be necessary for reasons of availability or performance, especially when the cache is poorly connected to the origin server. Whenever a cache returns a stale response, it MUST mark it as such (using a Warning header) enabling the client software to alert the user that there might be a potential problem.





It also allows the user agent to take steps to obtain a first-hand or fresh response. For this reason, a cache SHOULD NOT return a stale response if the client explicitly requests a first-hand or fresh one, unless it is impossible to comply for technical or policy reasons.

### 13.1.6 Client-controlled Behavior

While the origin server (and to a lesser extent, intermediate caches, by their contribution to the age of a response) are the primary source of expiration information, in some cases the client might need to control a cache's decision about whether to return a cached response without validating it. Clients do this using several directives of the Cache-Control header.

A client's request MAY specify the maximum age it is willing to accept of an unvalidated response; specifying a value of zero forces the cache(s) to revalidate all responses. A client MAY also specify the minimum time remaining before a response expires. Both of these options increase constraints on the behavior of caches, and so cannot further relax the cache's approximation of semantic transparency.

A client MAY also specify that it will accept stale responses, up to some maximum amount of staleness. This loosens the constraints on the caches, and so might violate the origin server's specified constraints on semantic transparency, but might be necessary to support disconnected operation, or high availability in the face of poor connectivity.

## 13.2 Expiration Model

### 13.2.1 Server-Specified Expiration

HTTP caching works best when caches can entirely avoid making requests to the origin server. The primary mechanism for avoiding requests is for an origin server to provide an explicit expiration

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

time in the future, indicating that a response MAY be used to satisfy subsequent requests. In other words, a cache can return a fresh response without first contacting the server.

Our expectation is that servers will assign future explicit expiration times to responses in the belief that the entity is not likely to change, in a semantically significant way, before the expiration time is reached. This normally preserves semantic transparency, as long as the server's expiration times are carefully chosen.





The expiration mechanism applies only to responses taken from a cache and not to first-hand responses forwarded immediately to the requesting client.

If an origin server wishes to force a semantically transparent cache to validate every request, it MAY assign an explicit expiration time in the past. This means that the response is always stale, and so the cache SHOULD validate it before using it for subsequent requests. See section 14.9.4 for a more restrictive way to force revalidation.

If an origin server wishes to force any HTTP/1.1 cache, no matter how it is configured, to validate every request, it SHOULD use the "must-revalidate" cache-control directive (see section 14.9).

Servers specify explicit expiration times using either the Expires header, or the max-age directive of the Cache-Control header.

An expiration time cannot be used to force a user agent to refresh its display or reload a resource; its semantics apply only to caching mechanisms, and such mechanisms need only check a resource's expiration status when a new request for that resource is initiated. See section 13.13 for an explanation of the difference between caches and history mechanisms.

### 13.2.2 Heuristic Expiration

Since origin servers do not always provide explicit expiration times, HTTP caches typically assign heuristic expiration times, employing algorithms that use other header values (such as the Last-Modified time) to estimate a plausible expiration time. The HTTP/1.1 specification does not provide specific algorithms, but does impose worst-case constraints on their results. Since heuristic expiration times might compromise semantic transparency, they ought to used cautiously, and we encourage origin servers to provide explicit expiration times as much as possible.

### 13.2.3 Age Calculations

In order to know if a cached entry is fresh, a cache needs to know if its age exceeds its freshness lifetime. We discuss how to calculate the latter in section 13.2.4; this section describes how to calculate the age of a response or cache entry.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

In this discussion, we use the term "now" to mean "the current value of the clock at the host performing the calculation." Hosts that use HTTP, but especially hosts running origin servers and caches, SHOULD use NTP [28] or some similar protocol to synchronize their clocks to a globally accurate time standard.





HTTP/1.1 requires origin servers to send a Date header, if possible, with every response, giving the time at which the response was generated (see section 14.18). We use the term "date_value" to denote the value of the Date header, in a form appropriate for arithmetic operations.

HTTP/1.1 uses the Age response-header to convey the estimated age of the response message when obtained from a cache. The Age field value is the cache's estimate of the amount of time since the response was generated or revalidated by the origin server.

In essence, the Age value is the sum of the time that the response has been resident in each of the caches along the path from the origin server, plus the amount of time it has been in transit along network paths.

We use the term "age_value" to denote the value of the Age header, in a form appropriate for arithmetic operations.

A response's age can be calculated in two entirely independent ways:

1. now minus date_value, if the local clock is reasonably well synchronized to the origin server's clock. If the result is negative, the result is replaced by zero.

2. age_value, if all of the caches along the response path implement HTTP/1.1.

Given that we have two independent ways to compute the age of a response when it is received, we can combine these as

```
corrected_received_age = max(now - date_value, age_value)
```

and as long as we have either nearly synchronized clocks or all-HTTP/1.1 paths, one gets a reliable (conservative) result.

Because of network-imposed delays, some significant interval might pass between the time that a server generates a response and the time it is received at the next outbound cache or client. If uncorrected, this delay could result in improperly low ages.

Because the request that resulted in the returned Age value must have been initiated prior to that Age value's generation, we can correct for delays imposed by the network by recording the time at which the request was initiated. Then, when an Age value is received, it MUST be interpreted relative to the time the request was initiated, not

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

the time that the response was received. This algorithm results in conservative behavior no matter how much delay is experienced. So, we compute:

```
corrected_initial_age = corrected_received_age
                      + (now - request_time)
```

where "request_time" is the time (according to the local clock) when the request that elicited this response was sent.

Summary of age calculation algorithm, when a cache receives a response:

```
/*
 * age_value
 *      is the value of Age: header received by the cache with
 *              this response.
 * date_value
 *      is the value of the origin server's Date: header
 * request_time
 *      is the (local) time when the cache made the request
 *              that resulted in this cached response
 * response_time
 *      is the (local) time when the cache received the
 *              response
 * now
 *      is the current (local) time
 */

apparent_age = max(0, response_time - date_value);
corrected_received_age = max(apparent_age, age_value);
response_delay = response_time - request_time;
corrected_initial_age = corrected_received_age + response_delay;
resident_time = now - response_time;
current_age   = corrected_initial_age + resident_time;
```

The current_age of a cache entry is calculated by adding the amount of time (in seconds) since the cache entry was last validated by the origin server to the corrected_initial_age. When a response is generated from a cache entry, the cache MUST include a single Age header field in the response with a value equal to the cache entry's current_age.

The presence of an Age header field in a response implies that a response is not first-hand. However, the converse is not true, since the lack of an Age header field in a response does not imply that the



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1150
Keysight, EX-1045
Page 1150

response is first-hand unless all caches along the request path are compliant with HTTP/1.1 (i.e., older HTTP caches did not implement the Age header field).

### 13.2.4 Expiration Calculations

In order to decide whether a response is fresh or stale, we need to compare its freshness lifetime to its age. The age is calculated as described in section 13.2.3; this section describes how to calculate the freshness lifetime, and to determine if a response has expired. In the discussion below, the values can be represented in any form appropriate for arithmetic operations.

We use the term "expires_value" to denote the value of the Expires header. We use the term "max_age_value" to denote an appropriate value of the number of seconds carried by the "max-age" directive of the Cache-Control header in a response (see section 14.9.3).

The max-age directive takes priority over Expires, so if max-age is present in a response, the calculation is simply:

```
freshness_lifetime = max_age_value
```

Otherwise, if Expires is present in the response, the calculation is:

```
freshness_lifetime = expires_value - date_value
```

Note that neither of these calculations is vulnerable to clock skew, since all of the information comes from the origin server.

If none of Expires, Cache-Control: max-age, or Cache-Control: s-maxage (see section 14.9.3) appears in the response, and the response does not include other restrictions on caching, the cache MAY compute a freshness lifetime using a heuristic. The cache MUST attach Warning 113 to any response whose age is more than 24 hours if such warning has not already been added.

Also, if the response does have a Last-Modified time, the heuristic expiration value SHOULD be no more than some fraction of the interval since that time. A typical setting of this fraction might be 10%.

The calculation to determine if a response has expired is quite simple:

```
response_is_fresh = (freshness_lifetime > current_age)
```





### 13.2.5 Disambiguating Expiration Values

Because expiration values are assigned optimistically, it is possible for two caches to contain fresh values for the same resource that are different.

If a client performing a retrieval receives a non-first-hand response

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1151
Keysight, EX-1045
Page 1151

for a request that was already fresh in its own cache, and the Date header in its existing cache entry is newer than the Date on the new response, then the client MAY ignore the response. If so, it MAY retry the request with a "Cache-Control: max-age=0" directive (see section 14.9), to force a check with the origin server.

If a cache has two fresh responses for the same representation with different validators, it MUST use the one with the more recent Date header. This situation might arise because the cache is pooling responses from other caches, or because a client has asked for a reload or a revalidation of an apparently fresh cache entry.

### 13.2.6 Disambiguating Multiple Responses

Because a client might be receiving responses via multiple paths, so that some responses flow through one set of caches and other responses flow through a different set of caches, a client might receive responses in an order different from that in which the origin server sent them. We would like the client to use the most recently generated response, even if older responses are still apparently fresh.

Neither the entity tag nor the expiration value can impose an ordering on responses, since it is possible that a later response intentionally carries an earlier expiration time. The Date values are ordered to a granularity of one second.

When a client tries to revalidate a cache entry, and the response it receives contains a Date header that appears to be older than the one for the existing entry, then the client SHOULD repeat the request unconditionally, and include

```
Cache-Control: max-age=0
```

to force any intermediate caches to validate their copies directly with the origin server, or

```
Cache-Control: no-cache
```

to force any intermediate caches to obtain a new copy from the origin server.





If the Date values are equal, then the client MAY use either response (or MAY, if it is being extremely prudent, request a new response). Servers MUST NOT depend on clients being able to choose deterministically between responses generated during the same second, if their expiration times overlap.

## 13.3 Validation Model

When a cache has a stale entry that it would like to use as a response to a client's request, it first has to check with the origin server (or possibly an intermediate cache with a fresh response) to see if its cached entry is still usable. We call this "validating" the cache entry. Since we do not want to have to pay the overhead of retransmitting the full response if the cached entry is good, and we

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1152
Keysight, EX-1045
Page 1152

do not want to pay the overhead of an extra round trip if the cached entry is invalid, the HTTP/1.1 protocol supports the use of conditional methods.

The key protocol features for supporting conditional methods are those concerned with "cache validators." When an origin server generates a full response, it attaches some sort of validator to it, which is kept with the cache entry. When a client (user agent or proxy cache) makes a conditional request for a resource for which it has a cache entry, it includes the associated validator in the request.

The server then checks that validator against the current validator for the entity, and, if they match (see section 13.3.3), it responds with a special status code (usually, 304 (Not Modified)) and no entity-body. Otherwise, it returns a full response (including entity-body). Thus, we avoid transmitting the full response if the validator matches, and we avoid an extra round trip if it does not match.

In HTTP/1.1, a conditional request looks exactly the same as a normal request for the same resource, except that it carries a special header (which includes the validator) that implicitly turns the method (usually, GET) into a conditional.

The protocol includes both positive and negative senses of cache-validating conditions. That is, it is possible to request either that a method be performed if and only if a validator matches or if and only if no validators match.





Note: a response that lacks a validator may still be cached, and served from cache until it expires, unless this is explicitly prohibited by a cache-control directive. However, a cache cannot do a conditional retrieval if it does not have a validator for the entity, which means it will not be refreshable after it expires.

### 13.3.1 Last-Modified Dates

The Last-Modified entity-header field value is often used as a cache validator. In simple terms, a cache entry is considered to be valid if the entity has not been modified since the Last-Modified value.

### 13.3.2 Entity Tag Cache Validators

The ETag response-header field value, an entity tag, provides for an "opaque" cache validator. This might allow more reliable validation in situations where it is inconvenient to store modification dates, where the one-second resolution of HTTP date values is not sufficient, or where the origin server wishes to avoid certain paradoxes that might arise from the use of modification dates.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1153
Keysight, EX-1045
Page 1153

Entity Tags are described in section 3.11. The headers used with entity tags are described in sections 14.19, 14.24, 14.26 and 14.44.

### 13.3.3 Weak and Strong Validators

Since both origin servers and caches will compare two validators to decide if they represent the same or different entities, one normally would expect that if the entity (the entity-body or any entity-headers) changes in any way, then the associated validator would change as well. If this is true, then we call this validator a "strong validator."

However, there might be cases when a server prefers to change the validator only on semantically significant changes, and not when insignificant aspects of the entity change. A validator that does not always change when the resource changes is a "weak validator."

Entity tags are normally "strong validators," but the protocol provides a mechanism to tag an entity tag as "weak." One can think of a strong validator as one that changes whenever the bits of an entity changes, while a weak value changes whenever the meaning of an entity changes. Alternatively, one can think of a strong validator as part of an identifier for a specific entity, while a weak validator is part of an identifier for a set of semantically equivalent entities.

> Note: One example of a strong validator is an integer that is incremented in stable storage every time an entity is changed.





> An entity's modification time, if represented with one-second resolution, could be a weak validator, since it is possible that the resource might be modified twice during a single second.
>
> Support for weak validators is optional. However, weak validators allow for more efficient caching of equivalent objects; for example, a hit counter on a site is probably good enough if it is updated every few days or weeks, and any value during that period is likely "good enough" to be equivalent.

A "use" of a validator is either when a client generates a request and includes the validator in a validating header field, or when a server compares two validators.

Strong validators are usable in any context. Weak validators are only usable in contexts that do not depend on exact equality of an entity. For example, either kind is usable for a conditional GET of a full entity. However, only a strong validator is usable for a sub-range retrieval, since otherwise the client might end up with an internally inconsistent entity.

Clients MAY issue simple (non-subrange) GET requests with either weak validators or strong validators. Clients MUST NOT use weak validators in other forms of request.

The only function that the HTTP/1.1 protocol defines on validators is comparison. There are two validator comparison functions, depending on whether the comparison context allows the use of weak validators

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

or not:

- The strong comparison function: in order to be considered equal, both validators MUST be identical in every way, and both MUST NOT be weak.

- The weak comparison function: in order to be considered equal, both validators MUST be identical in every way, but either or both of them MAY be tagged as "weak" without affecting the result.

An entity tag is strong unless it is explicitly tagged as weak. Section 3.11 gives the syntax for entity tags.

A Last-Modified time, when used as a validator in a request, is implicitly weak unless it is possible to deduce that it is strong, using the following rules:

- The validator is being compared by an origin server to the actual current validator for the entity and,





- That origin server reliably knows that the associated entity did not change twice during the second covered by the presented validator.

or

- The validator is about to be used by a client in an If-Modified-Since or If-Unmodified-Since header, because the client has a cache entry for the associated entity, and

- That cache entry includes a Date value, which gives the time when the origin server sent the original response, and

- The presented Last-Modified time is at least 60 seconds before the Date value.

or

- The validator is being compared by an intermediate cache to the validator stored in its cache entry for the entity, and

- That cache entry includes a Date value, which gives the time when the origin server sent the original response, and

- The presented Last-Modified time is at least 60 seconds before the Date value.

This method relies on the fact that if two different responses were sent by the origin server during the same second, but both had the same Last-Modified time, then at least one of those responses would have a Date value equal to its Last-Modified time. The arbitrary 60-second limit guards against the possibility that the Date and Last-Modified values are generated from different clocks, or at somewhat different times during the preparation of the response. An implementation MAY use a value larger than 60 seconds, if it is

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

believed that 60 seconds is too short.

If a client wishes to perform a sub-range retrieval on a value for which it has only a Last-Modified time and no opaque validator, it MAY do this only if the Last-Modified time is strong in the sense described here.

A cache or origin server receiving a conditional request, other than a full-body GET request, MUST use the strong comparison function to evaluate the condition.

These rules allow HTTP/1.1 caches and clients to safely perform sub-range retrievals on values that have been obtained from HTTP/1.0





servers.

### 13.3.4 Rules for When to Use Entity Tags and Last-Modified Dates

We adopt a set of rules and recommendations for origin servers, clients, and caches regarding when various validator types ought to be used, and for what purposes.

HTTP/1.1 origin servers:

- SHOULD send an entity tag validator unless it is not feasible to generate one.
- MAY send a weak entity tag instead of a strong entity tag, if performance considerations support the use of weak entity tags, or if it is unfeasible to send a strong entity tag.
- SHOULD send a Last-Modified value if it is feasible to send one, unless the risk of a breakdown in semantic transparency that could result from using this date in an If-Modified-Since header would lead to serious problems.

In other words, the preferred behavior for an HTTP/1.1 origin server is to send both a strong entity tag and a Last-Modified value.

In order to be legal, a strong entity tag MUST change whenever the associated entity value changes in any way. A weak entity tag SHOULD change whenever the associated entity changes in a semantically significant way.

> Note: in order to provide semantically transparent caching, an origin server must avoid reusing a specific strong entity tag value for two different entities, or reusing a specific weak entity tag value for two semantically different entities. Cache entries might persist for arbitrarily long periods, regardless of expiration times, so it might be inappropriate to expect that a cache will never again attempt to validate an entry using a validator that it obtained at some point in the past.

HTTP/1.1 clients:

- If an entity tag has been provided by the origin server, MUST

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1156
Keysight, EX-1045
Page 1156

use that entity tag in any cache-conditional request (using If-Match or If-None-Match).

- If only a Last-Modified value has been provided by the origin server, SHOULD use that value in non-subrange cache-conditional requests (using If-Modified-Since).





- If only a Last-Modified value has been provided by an HTTP/1.0 origin server, MAY use that value in subrange cache-conditional requests (using If-Unmodified-Since:). The user agent SHOULD provide a way to disable this, in case of difficulty.

- If both an entity tag and a Last-Modified value have been provided by the origin server, SHOULD use both validators in cache-conditional requests. This allows both HTTP/1.0 and HTTP/1.1 caches to respond appropriately.

An HTTP/1.1 origin server, upon receiving a conditional request that includes both a Last-Modified date (e.g., in an If-Modified-Since or If-Unmodified-Since header field) and one or more entity tags (e.g., in an If-Match, If-None-Match, or If-Range header field) as cache validators, MUST NOT return a response status of 304 (Not Modified) unless doing so is consistent with all of the conditional header fields in the request.

An HTTP/1.1 caching proxy, upon receiving a conditional request that includes both a Last-Modified date and one or more entity tags as cache validators, MUST NOT return a locally cached response to the client unless that cached response is consistent with all of the conditional header fields in the request.

Note: The general principle behind these rules is that HTTP/1.1 servers and clients should transmit as much non-redundant information as is available in their responses and requests. HTTP/1.1 systems receiving this information will make the most conservative assumptions about the validators they receive.

HTTP/1.0 clients and caches will ignore entity tags. Generally, last-modified values received or used by these systems will support transparent and efficient caching, and so HTTP/1.1 origin servers should provide Last-Modified values. In those rare cases where the use of a Last-Modified value as a validator by an HTTP/1.0 system could result in a serious problem, then HTTP/1.1 origin servers should not provide one.

### 13.3.5 Non-validating Conditionals

The principle behind entity tags is that only the service author knows the semantics of a resource well enough to select an appropriate cache validation mechanism, and the specification of any validator comparison function more complex than byte-equality would open up a can of worms. Thus, comparisons of any other headers (except Last-Modified, for compatibility with HTTP/1.0) are never used for purposes of validating a cache entry.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

## 13.4 Response Cacheability

Unless specifically constrained by a cache-control (section 14.9) directive, a caching system MAY always store a successful response (see section 13.8) as a cache entry, MAY return it without validation if it is fresh, and MAY return it after successful validation. If there is neither a cache validator nor an explicit expiration time associated with a response, we do not expect it to be cached, but certain caches MAY violate this expectation (for example, when little or no network connectivity is available). A client can usually detect that such a response was taken from a cache by comparing the Date header to the current time.

> Note: some HTTP/1.0 caches are known to violate this expectation without providing any Warning.

However, in some cases it might be inappropriate for a cache to retain an entity, or to return it in response to a subsequent request. This might be because absolute semantic transparency is deemed necessary by the service author, or because of security or privacy considerations. Certain cache-control directives are therefore provided so that the server can indicate that certain resource entities, or portions thereof, are not to be cached regardless of other considerations.

Note that section 14.8 normally prevents a shared cache from saving and returning a response to a previous request if that request included an Authorization header.

A response received with a status code of 200, 203, 206, 300, 301 or 410 MAY be stored by a cache and used in reply to a subsequent request, subject to the expiration mechanism, unless a cache-control directive prohibits caching. However, a cache that does not support the Range and Content-Range headers MUST NOT cache 206 (Partial Content) responses.

A response received with any other status code (e.g. status codes 302 and 307) MUST NOT be returned in a reply to a subsequent request unless there are cache-control directives or another header(s) that explicitly allow it. For example, these include the following: an Expires header (section 14.21); a "max-age", "s-maxage", "must-revalidate", "proxy-revalidate", "public" or "private" cache-control directive (section 14.9).



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

### 13.5 Constructing Responses From Caches

The purpose of an HTTP cache is to store information received in response to requests for use in responding to future requests. In many cases, a cache simply returns the appropriate parts of a response to the requester. However, if the cache holds a cache entry based on a previous response, it might have to combine parts of a new response with what is held in the cache entry.

#### 13.5.1 End-to-end and Hop-by-hop Headers

For the purpose of defining the behavior of caches and non-caching proxies, we divide HTTP headers into two categories:

- End-to-end headers, which are transmitted to the ultimate recipient of a request or response. End-to-end headers in responses MUST be stored as part of a cache entry and MUST be transmitted in any response formed from a cache entry.

- Hop-by-hop headers, which are meaningful only for a single transport-level connection, and are not stored by caches or forwarded by proxies.

The following HTTP/1.1 headers are hop-by-hop headers:

- Connection
- Keep-Alive
- Proxy-Authenticate
- Proxy-Authorization
- TE
- Trailers
- Transfer-Encoding
- Upgrade

All other headers defined by HTTP/1.1 are end-to-end headers.

Other hop-by-hop headers MUST be listed in a Connection header, (section 14.10) to be introduced into HTTP/1.1 (or later).

#### 13.5.2 Non-modifiable Headers

Some features of the HTTP/1.1 protocol, such as Digest Authentication, depend on the value of certain end-to-end headers. A transparent proxy SHOULD NOT modify an end-to-end header unless the definition of that header requires or specifically allows that.





A transparent proxy MUST NOT modify any of the following fields in a request or response, and it MUST NOT add any of these fields if not already present:

- Content-Location

- Content-MD5

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

- ETag

- Last-Modified

A transparent proxy MUST NOT modify any of the following fields in a response:

- Expires

but it MAY add any of these fields if not already present. If an Expires header is added, it MUST be given a field-value identical to that of the Date header in that response.

A proxy MUST NOT modify or add any of the following fields in a message that contains the no-transform cache-control directive, or in any request:

- Content-Encoding

- Content-Range

- Content-Type

A non-transparent proxy MAY modify or add these fields to a message that does not include no-transform, but if it does so, it MUST add a Warning 214 (Transformation applied) if one does not already appear in the message (see section 14.46).

> Warning: unnecessary modification of end-to-end headers might cause authentication failures if stronger authentication mechanisms are introduced in later versions of HTTP. Such authentication mechanisms MAY rely on the values of header fields not listed here.

The Content-Length field of a request or response is added or deleted according to the rules in section 4.4. A transparent proxy MUST preserve the entity-length (section 7.2.2) of the entity-body, although it MAY change the transfer-length (section 4.4).





### 13.5.3 Combining Headers

When a cache makes a validating request to a server, and the server provides a 304 (Not Modified) response or a 206 (Partial Content) response, the cache then constructs a response to send to the requesting client.

If the status code is 304 (Not Modified), the cache uses the entity-body stored in the cache entry as the entity-body of this outgoing response. If the status code is 206 (Partial Content) and the ETag or Last-Modified headers match exactly, the cache MAY combine the contents stored in the cache entry with the new contents received in the response and use the result as the entity-body of this outgoing response, (see 13.5.4).

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The end-to-end headers stored in the cache entry are used for the constructed response, except that

- any stored Warning headers with warn-code 1xx (see section 14.46) MUST be deleted from the cache entry and the forwarded response.

- any stored Warning headers with warn-code 2xx MUST be retained in the cache entry and the forwarded response.

- any end-to-end headers provided in the 304 or 206 response MUST replace the corresponding headers from the cache entry.

Unless the cache decides to remove the cache entry, it MUST also replace the end-to-end headers stored with the cache entry with corresponding headers received in the incoming response, except for Warning headers as described immediately above. If a header field-name in the incoming response matches more than one header in the cache entry, all such old headers MUST be replaced.

In other words, the set of end-to-end headers received in the incoming response overrides all corresponding end-to-end headers stored with the cache entry (except for stored Warning headers with warn-code 1xx, which are deleted even if not overridden).

Note: this rule allows an origin server to use a 304 (Not Modified) or a 206 (Partial Content) response to update any header associated with a previous response for the same entity or sub-ranges thereof, although it might not always be meaningful or correct to do so. This rule does not allow an origin server to use a 304 (Not Modified) or a 206 (Partial Content) response to entirely delete a header that it had provided with a previous response.





### 13.5.4 Combining Byte Ranges

A response might transfer only a subrange of the bytes of an entity-body, either because the request included one or more Range specifications, or because a connection was broken prematurely. After several such transfers, a cache might have received several ranges of the same entity-body.

If a cache has a stored non-empty set of subranges for an entity, and an incoming response transfers another subrange, the cache MAY combine the new subrange with the existing set if both the following conditions are met:

- Both the incoming response and the cache entry have a cache validator.

- The two cache validators match using the strong comparison function (see section 13.3.3).

If either requirement is not met, the cache MUST use only the most recent partial response (based on the Date values transmitted with

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1161
Keysight, EX-1045
Page 1161

every response, and using the incoming response if these values are equal or missing), and MUST discard the other partial information.

### 13.6 Caching Negotiated Responses

Use of server-driven content negotiation (section 12.1), as indicated by the presence of a Vary header field in a response, alters the conditions and procedure by which a cache can use the response for subsequent requests. See section 14.44 for use of the Vary header field by servers.

A server SHOULD use the Vary header field to inform a cache of what request-header fields were used to select among multiple representations of a cacheable response subject to server-driven negotiation. The set of header fields named by the Vary field value is known as the "selecting" request-headers.

When the cache receives a subsequent request whose Request-URI specifies one or more cache entries including a Vary header field, the cache MUST NOT use such a cache entry to construct a response to the new request unless all of the selecting request-headers present in the new request match the corresponding stored request-headers in the original request.

The selecting request-headers from two requests are defined to match if and only if the selecting request-headers in the first request can be transformed to the selecting request-headers in the second request





by adding or removing linear white space (LWS) at places where this is allowed by the corresponding BNF, and/or combining multiple message-header fields with the same field name following the rules about message headers in section 4.2.

A Vary header field-value of "*" always fails to match and subsequent requests on that resource can only be properly interpreted by the origin server.

If the selecting request header fields for the cached entry do not match the selecting request header fields of the new request, then the cache MUST NOT use a cached entry to satisfy the request unless it first relays the new request to the origin server in a conditional request and the server responds with 304 (Not Modified), including an entity tag or Content-Location that indicates the entity to be used.

If an entity tag was assigned to a cached representation, the forwarded request SHOULD be conditional and include the entity tags in an If-None-Match header field from all its cache entries for the resource. This conveys to the server the set of entities currently held by the cache, so that if any one of these entities matches the requested entity, the server can use the ETag header field in its 304 (Not Modified) response to tell the cache which entry is appropriate. If the entity-tag of the new response matches that of an existing entry, the new response SHOULD be used to update the header fields of the existing entry, and the result MUST be returned to the client.

If any of the existing cache entries contains only partial content

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

for the associated entity, its entity-tag SHOULD NOT be included in the If-None-Match header field unless the request is for a range that would be fully satisfied by that entry.

If a cache receives a successful response whose Content-Location field matches that of an existing cache entry for the same Request-]URI, whose entity-tag differs from that of the existing entry, and whose Date is more recent than that of the existing entry, the existing entry SHOULD NOT be returned in response to future requests and SHOULD be deleted from the cache.

13.7 Shared and Non-Shared Caches

For reasons of security and privacy, it is necessary to make a distinction between "shared" and "non-shared" caches. A non-shared cache is one that is accessible only to a single user. Accessibility in this case SHOULD be enforced by appropriate security mechanisms. All other caches are considered to be "shared." Other sections of





this specification place certain constraints on the operation of shared caches in order to prevent loss of privacy or failure of access controls.

13.8 Errors or Incomplete Response Cache Behavior

A cache that receives an incomplete response (for example, with fewer bytes of data than specified in a Content-Length header) MAY store the response. However, the cache MUST treat this as a partial response. Partial responses MAY be combined as described in section 13.5.4; the result might be a full response or might still be partial. A cache MUST NOT return a partial response to a client without explicitly marking it as such, using the 206 (Partial Content) status code. A cache MUST NOT return a partial response using a status code of 200 (OK).

If a cache receives a 5xx response while attempting to revalidate an entry, it MAY either forward this response to the requesting client, or act as if the server failed to respond. In the latter case, it MAY return a previously received response unless the cached entry includes the "must-revalidate" cache-control directive (see section 14.9).

13.9 Side Effects of GET and HEAD

Unless the origin server explicitly prohibits the caching of their responses, the application of GET and HEAD methods to any resources SHOULD NOT have side effects that would lead to erroneous behavior if these responses are taken from a cache. They MAY still have side effects, but a cache is not required to consider such side effects in its caching decisions. Caches are always expected to observe an origin server's explicit restrictions on caching.

We note one exception to this rule: since some applications have traditionally used GETs and HEADs with query URLs (those containing a

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1163
Keysight, EX-1045
Page 1163

"?" in the rel_path part) to perform operations with significant side effects, caches MUST NOT treat responses to such URIs as fresh unless the server provides an explicit expiration time. This specifically means that responses from HTTP/1.0 servers for such URIs SHOULD NOT be taken from a cache. See section 9.1.1 for related information.

### 13.10 Invalidation After Updates or Deletions

The effect of certain methods performed on a resource at the origin server might cause one or more existing cache entries to become non-transparently invalid. That is, although they might continue to be "fresh," they do not accurately reflect what the origin server would return for a new request on that resource.





There is no way for the HTTP protocol to guarantee that all such cache entries are marked invalid. For example, the request that caused the change at the origin server might not have gone through the proxy where a cache entry is stored. However, several rules help reduce the likelihood of erroneous behavior.

In this section, the phrase "invalidate an entity" means that the cache will either remove all instances of that entity from its storage, or will mark these as "invalid" and in need of a mandatory revalidation before they can be returned in response to a subsequent request.

Some HTTP methods MUST cause a cache to invalidate an entity. This is either the entity referred to by the Request-URI, or by the Location or Content-Location headers (if present). These methods are:

- PUT
- DELETE
- POST

In order to prevent denial of service attacks, an invalidation based on the URI in a Location or Content-Location header MUST only be performed if the host part is the same as in the Request-URI.

A cache that passes through requests for methods it does not understand SHOULD invalidate any entities referred to by the Request-URI.

### 13.11 Write-Through Mandatory

All methods that might be expected to cause modifications to the origin server's resources MUST be written through to the origin server. This currently includes all methods except for GET and HEAD. A cache MUST NOT reply to such a request from a client before having transmitted the request to the inbound server, and having received a corresponding response from the inbound server. This does not prevent a proxy cache from sending a 100 (Continue) response before the inbound server has sent its final reply.

The alternative (known as "write-back" or "copy-back" caching) is not

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

allowed in HTTP/1.1, due to the difficulty of providing consistent updates and the problems arising from server, cache, or network failure prior to write-back.





## 13.12 Cache Replacement

If a new cacheable (see sections 14.9.2, 13.2.5, 13.2.6 and 13.8) response is received from a resource while any existing responses for the same resource are cached, the cache SHOULD use the new response to reply to the current request. It MAY insert it into cache storage and MAY, if it meets all other requirements, use it to respond to any future requests that would previously have caused the old response to be returned. If it inserts the new response into cache storage the rules in section 13.5.3 apply.

Note: a new response that has an older Date header value than existing cached responses is not cacheable.

## 13.13 History Lists

User agents often have history mechanisms, such as "Back" buttons and history lists, which can be used to redisplay an entity retrieved earlier in a session.

History mechanisms and caches are different. In particular history mechanisms SHOULD NOT try to show a semantically transparent view of the current state of a resource. Rather, a history mechanism is meant to show exactly what the user saw at the time when the resource was retrieved.

By default, an expiration time does not apply to history mechanisms. If the entity is still in storage, a history mechanism SHOULD display it even if the entity has expired, unless the user has specifically configured the agent to refresh expired history documents.

This is not to be construed to prohibit the history mechanism from telling the user that a view might be stale.

Note: if history list mechanisms unnecessarily prevent users from viewing stale resources, this will tend to force service authors to avoid using HTTP expiration controls and cache controls when they would otherwise like to. Service authors may consider it important that users not be presented with error messages or warning messages when they use navigation controls (such as BACK) to view previously fetched resources. Even though sometimes such resources ought not to cached, or ought to expire quickly, user interface considerations may force service authors to resort to other means of preventing caching (e.g. "once-only" URLs) in order not to suffer the effects of improperly functioning history mechanisms.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

## 14 Header Field Definitions

This section defines the syntax and semantics of all standard HTTP/1.1 header fields. For entity-header fields, both sender and recipient refer to either the client or the server, depending on who sends and who receives the entity.

### 14.1 Accept

The Accept request-header field can be used to specify certain media types which are acceptable for the response. Accept headers can be used to indicate that the request is specifically limited to a small set of desired types, as in the case of a request for an in-line image.

```
    Accept         = "Accept" ":"
                     #( media-range [ accept-params ] )

    media-range    = ( "*/*"
                     | ( type "/" "*" )
                     | ( type "/" subtype )
                     ) *( ";" parameter )
    accept-params  = ";" "q" "=" qvalue *( accept-extension )
    accept-extension = ";" token [ "=" ( token | quoted-string ) ]
```

The asterisk "*" character is used to group media types into ranges, with "*/*" indicating all media types and "type/*" indicating all subtypes of that type. The media-range MAY include media type parameters that are applicable to that range.

Each media-range MAY be followed by one or more accept-params, beginning with the "q" parameter for indicating a relative quality factor. The first "q" parameter (if any) separates the media-range parameter(s) from the accept-params. Quality factors allow the user or user agent to indicate the relative degree of preference for that media-range, using the qvalue scale from 0 to 1 (section 3.9). The default value is q=1.

> Note: Use of the "q" parameter name to separate media type parameters from Accept extension parameters is due to historical practice. Although this prevents any media type parameter named "q" from being used with a media range, such an event is believed to be unlikely given the lack of any "q" parameters in the IANA media type registry and the rare usage of any media type parameters in Accept. Future media types are discouraged from registering any parameter named "q".



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The example

```
    Accept: audio/*; q=0.2, audio/basic
```

SHOULD be interpreted as "I prefer audio/basic, but send me any audio type if it is the best available after an 80% mark-down in quality."

If no Accept header field is present, then it is assumed that the client accepts all media types. If an Accept header field is present, and if the server cannot send a response which is acceptable according to the combined Accept field value, then the server SHOULD send a 406 (not acceptable) response.

A more elaborate example is

```
    Accept: text/plain; q=0.5, text/html,
            text/x-dvi; q=0.8, text/x-c
```

Verbally, this would be interpreted as "text/html and text/x-c are the preferred media types, but if they do not exist, then send the text/x-dvi entity, and if that does not exist, send the text/plain entity."

Media ranges can be overridden by more specific media ranges or specific media types. If more than one media range applies to a given type, the most specific reference has precedence. For example,

```
    Accept: text/*, text/html, text/html;level=1, */*
```

have the following precedence:

1) text/html;level=1
2) text/html
3) text/*
4) */*

The media type quality factor associated with a given type is determined by finding the media range with the highest precedence which matches that type. For example,

```
    Accept: text/*;q=0.3, text/html;q=0.7, text/html;level=1,
            text/html;level=2;q=0.4, */*;q=0.5
```

would cause the following values to be associated:

```
    text/html;level=1         = 1
    text/html                 = 0.7
    text/plain                = 0.3
```





```
    image/jpeg                = 0.5
    text/html;level=2         = 0.4
    text/html;level=3         = 0.7
```

Note: A user agent might be provided with a default set of quality values for certain media ranges. However, unless the user agent is a closed system which cannot interact with other rendering agents,

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

this default set ought to be configurable by the user.

### 14.2 Accept-Charset

The Accept-Charset request-header field can be used to indicate what character sets are acceptable for the response. This field allows clients capable of understanding more comprehensive or special-purpose character sets to signal that capability to a server which is capable of representing documents in those character sets.

```
Accept-Charset = "Accept-Charset" ":"
        1#( ( charset | "*" )[ ";" "q" "=" qvalue ] )
```

Character set values are described in section 3.4. Each charset MAY be given an associated quality value which represents the user's preference for that charset. The default value is q=1. An example is

```
Accept-Charset: iso-8859-5, unicode-1-1;q=0.8
```

The special value "*", if present in the Accept-Charset field, matches every character set (including ISO-8859-1) which is not mentioned elsewhere in the Accept-Charset field. If no "*" is present in an Accept-Charset field, then all character sets not explicitly mentioned get a quality value of 0, except for ISO-8859-1, which gets a quality value of 1 if not explicitly mentioned.

If no Accept-Charset header is present, the default is that any character set is acceptable. If an Accept-Charset header is present, and if the server cannot send a response which is acceptable according to the Accept-Charset header, then the server SHOULD send an error response with the 406 (not acceptable) status code, though the sending of an unacceptable response is also allowed.

### 14.3 Accept-Encoding

The Accept-Encoding request-header field is similar to Accept, but restricts the content-codings (section 3.5) that are acceptable in the response.

```
Accept-Encoding  = "Accept-Encoding" ":"
```





```
                      1#( codings [ ";" "q" "=" qvalue ] )
codings          = ( content-coding | "*" )
```

Examples of its use are:

```
Accept-Encoding: compress, gzip
Accept-Encoding:
Accept-Encoding: *
Accept-Encoding: compress;q=0.5, gzip;q=1.0
Accept-Encoding: gzip;q=1.0, identity; q=0.5, *;q=0
```

A server tests whether a content-coding is acceptable, according to an Accept-Encoding field, using these rules:

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

1. If the content-coding is one of the content-codings listed in the Accept-Encoding field, then it is acceptable, unless it is accompanied by a qvalue of 0. (As defined in section 3.9, a qvalue of 0 means "not acceptable.")

2. The special "*" symbol in an Accept-Encoding field matches any available content-coding not explicitly listed in the header field.

3. If multiple content-codings are acceptable, then the acceptable content-coding with the highest non-zero qvalue is preferred.

4. The "identity" content-coding is always acceptable, unless specifically refused because the Accept-Encoding field includes "identity;q=0", or because the field includes "*;q=0" and does not explicitly include the "identity" content-coding. If the Accept-Encoding field-value is empty, then only the "identity" encoding is acceptable.

If an Accept-Encoding field is present in a request, and if the server cannot send a response which is acceptable according to the Accept-Encoding header, then the server SHOULD send an error response with the 406 (Not Acceptable) status code.

If no Accept-Encoding field is present in a request, the server MAY assume that the client will accept any content coding. In this case, if "identity" is one of the available content-codings, then the server SHOULD use the "identity" content-coding, unless it has additional information that a different content-coding is meaningful to the client.

Note: If the request does not include an Accept-Encoding field, and if the "identity" content-coding is unavailable, then content-codings commonly understood by HTTP/1.0 clients (i.e.,





"gzip" and "compress") are preferred; some older clients improperly display messages sent with other content-codings. The server might also make this decision based on information about the particular user-agent or client.

Note: Most HTTP/1.0 applications do not recognize or obey qvalues associated with content-codings. This means that qvalues will not work and are not permitted with x-gzip or x-compress.

### 14.4 Accept-Language

The Accept-Language request-header field is similar to Accept, but restricts the set of natural languages that are preferred as a response to the request. Language tags are defined in section 3.10.

```
Accept-Language = "Accept-Language" ":"
                  1#( language-range [ ";" "q" "=" qvalue ] )
language-range  = ( ( 1*8ALPHA *( "-" 1*8ALPHA ) ) | "*" )
```

Each language-range MAY be given an associated quality value which represents an estimate of the user's preference for the languages

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

specified by that range. The quality value defaults to "q=1". For example,

```
    Accept-Language: da, en-gb;q=0.8, en;q=0.7
```

would mean: "I prefer Danish, but will accept British English and other types of English." A language-range matches a language-tag if it exactly equals the tag, or if it exactly equals a prefix of the tag such that the first tag character following the prefix is "-". The special range "*", if present in the Accept-Language field, matches every tag not matched by any other range present in the Accept-Language field.

> Note: This use of a prefix matching rule does not imply that language tags are assigned to languages in such a way that it is always true that if a user understands a language with a certain tag, then this user will also understand all languages with tags for which this tag is a prefix. The prefix rule simply allows the use of prefix tags if this is the case.

The language quality factor assigned to a language-tag by the Accept-Language field is the quality value of the longest language-range in the field that matches the language-tag. If no language-range in the field matches the tag, the language quality factor assigned is 0. If no Accept-Language header is present in the request, the server





SHOULD assume that all languages are equally acceptable. If an Accept-Language header is present, then all languages which are assigned a quality factor greater than 0 are acceptable.

It might be contrary to the privacy expectations of the user to send an Accept-Language header with the complete linguistic preferences of the user in every request. For a discussion of this issue, see section 15.1.4.

As intelligibility is highly dependent on the individual user, it is recommended that client applications make the choice of linguistic preference available to the user. If the choice is not made available, then the Accept-Language header field MUST NOT be given in the request.

> Note: When making the choice of linguistic preference available to the user, we remind implementors of the fact that users are not familiar with the details of language matching as described above, and should provide appropriate guidance. As an example, users might assume that on selecting "en-gb", they will be served any kind of English document if British English is not available. A user agent might suggest in such a case to add "en" to get the best matching behavior.

### 14.5 Accept-Ranges

The Accept-Ranges response-header field allows the server to indicate its acceptance of range requests for a resource:

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

```
    Accept-Ranges     = "Accept-Ranges" ":" acceptable-ranges
    acceptable-ranges = 1#range-unit | "none"
```

Origin servers that accept byte-range requests MAY send

```
    Accept-Ranges: bytes
```

but are not required to do so. Clients MAY generate byte-range requests without having received this header for the resource involved. Range units are defined in section 3.12.

Servers that do not accept any kind of range request for a resource MAY send

```
    Accept-Ranges: none
```

to advise the client not to attempt a range request.





## 14.6 Age

The Age response-header field conveys the sender's estimate of the amount of time since the response (or its revalidation) was generated at the origin server. A cached response is "fresh" if its age does not exceed its freshness lifetime. Age values are calculated as specified in section 13.2.3.

```
     Age = "Age" ":" age-value
     age-value = delta-seconds
```

Age values are non-negative decimal integers, representing time in seconds.

If a cache receives a value larger than the largest positive integer it can represent, or if any of its age calculations overflows, it MUST transmit an Age header with a value of 2147483648 (2^31). An HTTP/1.1 server that includes a cache MUST include an Age header field in every response generated from its own cache. Caches SHOULD use an arithmetic type of at least 31 bits of range.

## 14.7 Allow

The Allow entity-header field lists the set of methods supported by the resource identified by the Request-URI. The purpose of this field is strictly to inform the recipient of valid methods associated with the resource. An Allow header field MUST be present in a 405 (Method Not Allowed) response.

```
    Allow   = "Allow" ":" #Method
```

Example of use:

```
    Allow: GET, HEAD, PUT
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

This field cannot prevent a client from trying other methods. However, the indications given by the Allow header field value SHOULD be followed. The actual set of allowed methods is defined by the origin server at the time of each request.

The Allow header field MAY be provided with a PUT request to recommend the methods to be supported by the new or modified resource. The server is not required to support these methods and SHOULD include an Allow header in the response giving the actual supported methods.





A proxy MUST NOT modify the Allow header field even if it does not understand all the methods specified, since the user agent might have other means of communicating with the origin server.

## 14.8 Authorization

A user agent that wishes to authenticate itself with a server--usually, but not necessarily, after receiving a 401 response--does so by including an Authorization request-header field with the request. The Authorization field value consists of credentials containing the authentication information of the user agent for the realm of the resource being requested.

```
Authorization  = "Authorization" ":" credentials
```

HTTP access authentication is described in "HTTP Authentication: Basic and Digest Access Authentication" [43]. If a request is authenticated and a realm specified, the same credentials SHOULD be valid for all other requests within this realm (assuming that the authentication scheme itself does not require otherwise, such as credentials that vary according to a challenge value or using synchronized clocks).

When a shared cache (see section 13.7) receives a request containing an Authorization field, it MUST NOT return the corresponding response as a reply to any other request, unless one of the following specific exceptions holds:

1. If the response includes the "s-maxage" cache-control directive, the cache MAY use that response in replying to a subsequent request. But (if the specified maximum age has passed) a proxy cache MUST first revalidate it with the origin server, using the request-headers from the new request to allow the origin server to authenticate the new request. (This is the defined behavior for s-maxage.) If the response includes "s-maxage=0", the proxy MUST always revalidate it before re-using it.

2. If the response includes the "must-revalidate" cache-control directive, the cache MAY use that response in replying to a subsequent request. But if the response is stale, all caches MUST first revalidate it with the origin server, using the

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1172
Keysight, EX-1045
Page 1172

request-headers from the new request to allow the origin server to authenticate the new request.

3. If the response includes the "public" cache-control directive, it MAY be returned in reply to any subsequent request.





## 14.9 Cache-Control

The Cache-Control general-header field is used to specify directives that MUST be obeyed by all caching mechanisms along the request/response chain. The directives specify behavior intended to prevent caches from adversely interfering with the request or response. These directives typically override the default caching algorithms. Cache directives are unidirectional in that the presence of a directive in a request does not imply that the same directive is to be given in the response.

Note that HTTP/1.0 caches might not implement Cache-Control and might only implement Pragma: no-cache (see section 14.32).

Cache directives MUST be passed through by a proxy or gateway application, regardless of their significance to that application, since the directives might be applicable to all recipients along the request/response chain. It is not possible to specify a cache-directive for a specific cache.

```
Cache-Control   = "Cache-Control" ":" 1#cache-directive

cache-directive = cache-request-directive
     | cache-response-directive

cache-request-directive =
       "no-cache"                          ; Section 14.9.1
     | "no-store"                          ; Section 14.9.2
     | "max-age" "=" delta-seconds         ; Section 14.9.3, 14.9.4
     | "max-stale" [ "=" delta-seconds ]   ; Section 14.9.3
     | "min-fresh" "=" delta-seconds       ; Section 14.9.3
     | "no-transform"                      ; Section 14.9.5
     | "only-if-cached"                    ; Section 14.9.4
     | cache-extension                     ; Section 14.9.6

 cache-response-directive =
       "public"                               ; Section 14.9.1
     | "private" [ "=" <"> 1#field-name <"> ] ; Section 14.9.1
     | "no-cache" [ "=" <"> 1#field-name <"> ]; Section 14.9.1
     | "no-store"                             ; Section 14.9.2
     | "no-transform"                         ; Section 14.9.5
     | "must-revalidate"                      ; Section 14.9.4
     | "proxy-revalidate"                     ; Section 14.9.4
     | "max-age" "=" delta-seconds            ; Section 14.9.3
     | "s-maxage" "=" delta-seconds           ; Section 14.9.3
     | cache-extension                        ; Section 14.9.6

cache-extension = token [ "=" ( token | quoted-string ) ]
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

When a directive appears without any 1#field-name parameter, the directive applies to the entire request or response. When such a directive appears with a 1#field-name parameter, it applies only to the named field or fields, and not to the rest of the request or response. This mechanism supports extensibility; implementations of future versions of the HTTP protocol might apply these directives to header fields not defined in HTTP/1.1.

The cache-control directives can be broken down into these general categories:

- Restrictions on what are cacheable; these may only be imposed by the origin server.
- Restrictions on what may be stored by a cache; these may be imposed by either the origin server or the user agent.
- Modifications of the basic expiration mechanism; these may be imposed by either the origin server or the user agent.
- Controls over cache revalidation and reload; these may only be imposed by a user agent.
- Control over transformation of entities.
- Extensions to the caching system.

### 14.9.1 What is Cacheable

By default, a response is cacheable if the requirements of the request method, request header fields, and the response status indicate that it is cacheable. Section 13.4 summarizes these defaults for cacheability. The following Cache-Control response directives allow an origin server to override the default cacheability of a response:

public
: Indicates that the response MAY be cached by any cache, even if it would normally be non-cacheable or cacheable only within a non-shared cache. (See also Authorization, section 14.8, for additional details.)

private
: Indicates that all or part of the response message is intended for a single user and MUST NOT be cached by a shared cache. This allows an origin server to state that the specified parts of the



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

response are intended for only one user and are not a valid response for requests by other users. A private (non-shared) cache MAY cache the response.

Note: This usage of the word private only controls where the response may be cached, and cannot ensure the privacy of the message content.

no-cache
If the no-cache directive does not specify a field-name, then a cache MUST NOT use the response to satisfy a subsequent request without successful revalidation with the origin server. This allows an origin server to prevent caching even by caches that have been configured to return stale responses to client requests.

If the no-cache directive does specify one or more field-names, then a cache MAY use the response to satisfy a subsequent request, subject to any other restrictions on caching. However, the specified field-name(s) MUST NOT be sent in the response to a subsequent request without successful revalidation with the origin server. This allows an origin server to prevent the re-use of certain header fields in a response, while still allowing caching of the rest of the response.

Note: Most HTTP/1.0 caches will not recognize or obey this directive.

14.9.2 What May be Stored by Caches

no-store
The purpose of the no-store directive is to prevent the inadvertent release or retention of sensitive information (for example, on backup tapes). The no-store directive applies to the entire message, and MAY be sent either in a response or in a request. If sent in a request, a cache MUST NOT store any part of either this request or any response to it. If sent in a response, a cache MUST NOT store any part of either this response or the request that elicited it. This directive applies to both non-shared and shared caches. "MUST NOT store" in this context means that the cache MUST NOT intentionally store the information in non-volatile storage, and MUST make a best-effort attempt to remove the information from volatile storage as promptly as possible after forwarding it.

Even when this directive is associated with a response, users might explicitly store such a response outside of the caching system (e.g., with a "Save As" dialog). History buffers MAY store such responses as part of their normal operation.





The purpose of this directive is to meet the stated requirements of certain users and service authors who are concerned about accidental releases of information via unanticipated accesses to cache data structures. While the use of this directive might improve privacy in some cases, we caution that it is NOT in any way a reliable or sufficient mechanism for ensuring privacy. In particular, malicious or compromised caches might not recognize or

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1175
Keysight, EX-1045
Page 1175

obey this directive, and communications networks might be vulnerable to eavesdropping.

### 14.9.3 Modifications of the Basic Expiration Mechanism

The expiration time of an entity MAY be specified by the origin server using the Expires header (see section 14.21). Alternatively, it MAY be specified using the max-age directive in a response. When the max-age cache-control directive is present in a cached response, the response is stale if its current age is greater than the age value given (in seconds) at the time of a new request for that resource. The max-age directive on a response implies that the response is cacheable (i.e., "public") unless some other, more restrictive cache directive is also present.

If a response includes both an Expires header and a max-age directive, the max-age directive overrides the Expires header, even if the Expires header is more restrictive. This rule allows an origin server to provide, for a given response, a longer expiration time to an HTTP/1.1 (or later) cache than to an HTTP/1.0 cache. This might be useful if certain HTTP/1.0 caches improperly calculate ages or expiration times, perhaps due to desynchronized clocks.

Many HTTP/1.0 cache implementations will treat an Expires value that is less than or equal to the response Date value as being equivalent to the Cache-Control response directive "no-cache". If an HTTP/1.1 cache receives such a response, and the response does not include a Cache-Control header field, it SHOULD consider the response to be non-cacheable in order to retain compatibility with HTTP/1.0 servers.

> Note: An origin server might wish to use a relatively new HTTP cache control feature, such as the "private" directive, on a network including older caches that do not understand that feature. The origin server will need to combine the new feature with an Expires field whose value is less than or equal to the Date value. This will prevent older caches from improperly caching the response.





s-maxage
: If a response includes an s-maxage directive, then for a shared cache (but not for a private cache), the maximum age specified by this directive overrides the maximum age specified by either the max-age directive or the Expires header. The s-maxage directive also implies the semantics of the proxy-revalidate directive (see section 14.9.4), i.e., that the shared cache must not use the entry after it becomes stale to respond to a subsequent request without first revalidating it with the origin server. The s-maxage directive is always ignored by a private cache.

Note that most older caches, not compliant with this specification, do not implement any cache-control directives. An origin server wishing to use a cache-control directive that restricts, but does not

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

prevent, caching by an HTTP/1.1-compliant cache MAY exploit the requirement that the max-age directive overrides the Expires header, and the fact that pre-HTTP/1.1-compliant caches do not observe the max-age directive.

Other directives allow a user agent to modify the basic expiration mechanism. These directives MAY be specified on a request:

max-age
Indicates that the client is willing to accept a response whose age is no greater than the specified time in seconds. Unless max-stale directive is also included, the client is not willing to accept a stale response.

min-fresh
Indicates that the client is willing to accept a response whose freshness lifetime is no less than its current age plus the specified time in seconds. That is, the client wants a response that will still be fresh for at least the specified number of seconds.

max-stale
Indicates that the client is willing to accept a response that has exceeded its expiration time. If max-stale is assigned a value, then the client is willing to accept a response that has exceeded its expiration time by no more than the specified number of seconds. If no value is assigned to max-stale, then the client is willing to accept a stale response of any age.

If a cache returns a stale response, either because of a max-stale directive on a request, or because the cache is configured to override the expiration time of a response, the cache MUST attach a Warning header to the stale response, using Warning 110 (Response is stale).





A cache MAY be configured to return stale responses without validation, but only if this does not conflict with any "MUST"-level requirements concerning cache validation (e.g., a "must-revalidate" cache-control directive).

If both the new request and the cached entry include "max-age" directives, then the lesser of the two values is used for determining the freshness of the cached entry for that request.

### 14.9.4 Cache Revalidation and Reload Controls

Sometimes a user agent might want or need to insist that a cache revalidate its cache entry with the origin server (and not just with the next cache along the path to the origin server), or to reload its cache entry from the origin server. End-to-end revalidation might be necessary if either the cache or the origin server has overestimated the expiration time of the cached response. End-to-end reload may be necessary if the cache entry has become corrupted for some reason.

End-to-end revalidation may be requested either when the client does not have its own local cached copy, in which case we call it

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

"unspecified end-to-end revalidation", or when the client does have a local cached copy, in which case we call it "specific end-to-end revalidation."

The client can specify these three kinds of action using Cache-Control request directives:

End-to-end reload
The request includes a "no-cache" cache-control directive or, for compatibility with HTTP/1.0 clients, "Pragma: no-cache". Field names MUST NOT be included with the no-cache directive in a request. The server MUST NOT use a cached copy when responding to such a request.

Specific end-to-end revalidation
The request includes a "max-age=0" cache-control directive, which forces each cache along the path to the origin server to revalidate its own entry, if any, with the next cache or server. The initial request includes a cache-validating conditional with the client's current validator.

Unspecified end-to-end revalidation
The request includes "max-age=0" cache-control directive, which forces each cache along the path to the origin server to revalidate its own entry, if any, with the next cache or server. The initial request does not include a cache-validating





conditional; the first cache along the path (if any) that holds a cache entry for this resource includes a cache-validating conditional with its current validator.

max-age
When an intermediate cache is forced, by means of a max-age=0 directive, to revalidate its own cache entry, and the client has supplied its own validator in the request, the supplied validator might differ from the validator currently stored with the cache entry. In this case, the cache MAY use either validator in making its own request without affecting semantic transparency.

However, the choice of validator might affect performance. The best approach is for the intermediate cache to use its own validator when making its request. If the server replies with 304 (Not Modified), then the cache can return its now validated copy to the client with a 200 (OK) response. If the server replies with a new entity and cache validator, however, the intermediate cache can compare the returned validator with the one provided in the client's request, using the strong comparison function. If the client's validator is equal to the origin server's, then the intermediate cache simply returns 304 (Not Modified). Otherwise, it returns the new entity with a 200 (OK) response.

If a request includes the no-cache directive, it SHOULD NOT include min-fresh, max-stale, or max-age.

only-if-cached

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

In some cases, such as times of extremely poor network connectivity, a client may want a cache to return only those responses that it currently has stored, and not to reload or revalidate with the origin server. To do this, the client may include the only-if-cached directive in a request. If it receives this directive, a cache SHOULD either respond using a cached entry that is consistent with the other constraints of the request, or respond with a 504 (Gateway Timeout) status. However, if a group of caches is being operated as a unified system with good internal connectivity, such a request MAY be forwarded within that group of caches.

must-revalidate
Because a cache MAY be configured to ignore a server's specified expiration time, and because a client request MAY include a max-stale directive (which has a similar effect), the protocol also includes a mechanism for the origin server to require revalidation of a cache entry on any subsequent use. When the must-revalidate directive is present in a response received by a cache, that cache MUST NOT use the entry after it becomes stale to respond to a





subsequent request without first revalidating it with the origin server. (I.e., the cache MUST do an end-to-end revalidation every time, if, based solely on the origin server's Expires or max-age value, the cached response is stale.)

The must-revalidate directive is necessary to support reliable operation for certain protocol features. In all circumstances an HTTP/1.1 cache MUST obey the must-revalidate directive; in particular, if the cache cannot reach the origin server for any reason, it MUST generate a 504 (Gateway Timeout) response.

Servers SHOULD send the must-revalidate directive if and only if failure to revalidate a request on the entity could result in incorrect operation, such as a silently unexecuted financial transaction. Recipients MUST NOT take any automated action that violates this directive, and MUST NOT automatically provide an unvalidated copy of the entity if revalidation fails.

Although this is not recommended, user agents operating under severe connectivity constraints MAY violate this directive but, if so, MUST explicitly warn the user that an unvalidated response has been provided. The warning MUST be provided on each unvalidated access, and SHOULD require explicit user confirmation.

proxy-revalidate
The proxy-revalidate directive has the same meaning as the must-revalidate directive, except that it does not apply to non-shared user agent caches. It can be used on a response to an authenticated request to permit the user's cache to store and later return the response without needing to revalidate it (since it has already been authenticated once by that user), while still requiring proxies that service many users to revalidate each time (in order to make sure that each user has been authenticated). Note that such authenticated responses also need the public cache control directive in order to allow them to be cached at all.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1179
Keysight, EX-1045
Page 1179

14.9.5 No-Transform Directive

no-transform
Implementors of intermediate caches (proxies) have found it useful to convert the media type of certain entity bodies. A non-transparent proxy might, for example, convert between image formats in order to save cache space or to reduce the amount of traffic on a slow link.

Serious operational problems occur, however, when these transformations are applied to entity bodies intended for certain kinds of applications. For example, applications for medical





imaging, scientific data analysis and those using end-to-end authentication, all depend on receiving an entity body that is bit for bit identical to the original entity-body.

Therefore, if a message includes the no-transform directive, an intermediate cache or proxy MUST NOT change those headers that are listed in section 13.5.2 as being subject to the no-transform directive. This implies that the cache or proxy MUST NOT change any aspect of the entity-body that is specified by these headers, including the value of the entity-body itself.

14.9.6 Cache Control Extensions

The Cache-Control header field can be extended through the use of one or more cache-extension tokens, each with an optional assigned value. Informational extensions (those which do not require a change in cache behavior) MAY be added without changing the semantics of other directives. Behavioral extensions are designed to work by acting as modifiers to the existing base of cache directives. Both the new directive and the standard directive are supplied, such that applications which do not understand the new directive will default to the behavior specified by the standard directive, and those that understand the new directive will recognize it as modifying the requirements associated with the standard directive. In this way, extensions to the cache-control directives can be made without requiring changes to the base protocol.

This extension mechanism depends on an HTTP cache obeying all of the cache-control directives defined for its native HTTP-version, obeying certain extensions, and ignoring all directives that it does not understand.

For example, consider a hypothetical new response directive called community which acts as a modifier to the private directive. We define this new directive to mean that, in addition to any non-shared cache, any cache which is shared only by members of the community named within its value may cache the response. An origin server wishing to allow the UCI community to use an otherwise private response in their shared cache(s) could do so by including

```
Cache-Control: private, community="UCI"
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

A cache seeing this header field will act correctly even if the cache does not understand the community cache-extension, since it will also see and understand the private directive and thus default to the safe behavior.





Unrecognized cache-directives MUST be ignored; it is assumed that any cache-directive likely to be unrecognized by an HTTP/1.1 cache will be combined with standard directives (or the response's default cacheability) such that the cache behavior will remain minimally correct even if the cache does not understand the extension(s).

### 14.10 Connection

The Connection general-header field allows the sender to specify options that are desired for that particular connection and MUST NOT be communicated by proxies over further connections.

The Connection header has the following grammar:

```
    Connection = "Connection" ":" 1#(connection-token)
    connection-token  = token
```

HTTP/1.1 proxies MUST parse the Connection header field before a message is forwarded and, for each connection-token in this field, remove any header field(s) from the message with the same name as the connection-token. Connection options are signaled by the presence of a connection-token in the Connection header field, not by any corresponding additional header field(s), since the additional header field may not be sent if there are no parameters associated with that connection option.

Message headers listed in the Connection header MUST NOT include end-to-end headers, such as Cache-Control.

HTTP/1.1 defines the "close" connection option for the sender to signal that the connection will be closed after completion of the response. For example,

```
    Connection: close
```

in either the request or the response header fields indicates that the connection SHOULD NOT be considered `persistent' (section 8.1) after the current request/response is complete.

HTTP/1.1 applications that do not support persistent connections MUST include the "close" connection option in every message.

A system receiving an HTTP/1.0 (or lower-version) message that includes a Connection header MUST, for each connection-token in this field, remove and ignore any header field(s) from the message with the same name as the connection-token. This protects against mistaken forwarding of such header fields by pre-HTTP/1.1 proxies. See section 19.6.2.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

### 14.11 Content-Encoding

The Content-Encoding entity-header field is used as a modifier to the media-type. When present, its value indicates what additional content codings have been applied to the entity-body, and thus what decoding mechanisms must be applied in order to obtain the media-type referenced by the Content-Type header field. Content-Encoding is primarily used to allow a document to be compressed without losing the identity of its underlying media type.

```
Content-Encoding  = "Content-Encoding" ":" 1#content-coding
```

Content codings are defined in section 3.5. An example of its use is

```
Content-Encoding: gzip
```

The content-coding is a characteristic of the entity identified by the Request-URI. Typically, the entity-body is stored with this encoding and is only decoded before rendering or analogous usage. However, a non-transparent proxy MAY modify the content-coding if the new coding is known to be acceptable to the recipient, unless the "no-transform" cache-control directive is present in the message.

If the content-coding of an entity is not "identity", then the response MUST include a Content-Encoding entity-header (section 14.11) that lists the non-identity content-coding(s) used.

If the content-coding of an entity in a request message is not acceptable to the origin server, the server SHOULD respond with a status code of 415 (Unsupported Media Type).

If multiple encodings have been applied to an entity, the content codings MUST be listed in the order in which they were applied. Additional information about the encoding parameters MAY be provided by other entity-header fields not defined by this specification.

### 14.12 Content-Language

The Content-Language entity-header field describes the natural language(s) of the intended audience for the enclosed entity. Note that this might not be equivalent to all the languages used within the entity-body.

```
Content-Language  = "Content-Language" ":" 1#language-tag
```



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1182
Keysight, EX-1045
Page 1182

Language tags are defined in section 3.10. The primary purpose of Content-Language is to allow a user to identify and differentiate entities according to the user's own preferred language. Thus, if the body content is intended only for a Danish-literate audience, the appropriate field is

```
    Content-Language: da
```

If no Content-Language is specified, the default is that the content is intended for all language audiences. This might mean that the sender does not consider it to be specific to any natural language, or that the sender does not know for which language it is intended.

Multiple languages MAY be listed for content that is intended for multiple audiences. For example, a rendition of the "Treaty of Waitangi," presented simultaneously in the original Maori and English versions, would call for

```
    Content-Language: mi, en
```

However, just because multiple languages are present within an entity does not mean that it is intended for multiple linguistic audiences. An example would be a beginner's language primer, such as "A First Lesson in Latin," which is clearly intended to be used by an English-literate audience. In this case, the Content-Language would properly only include "en".

Content-Language MAY be applied to any media type -- it is not limited to textual documents.

### 14.13 Content-Length

The Content-Length entity-header field indicates the size of the entity-body, in decimal number of OCTETs, sent to the recipient or, in the case of the HEAD method, the size of the entity-body that would have been sent had the request been a GET.

```
    Content-Length    = "Content-Length" ":" 1*DIGIT
```

An example is

```
    Content-Length: 3495
```

Applications SHOULD use this field to indicate the transfer-length of the message-body, unless this is prohibited by the rules in section 4.4.





Any Content-Length greater than or equal to zero is a valid value. Section 4.4 describes how to determine the length of a message-body if a Content-Length is not given.

Note that the meaning of this field is significantly different from the corresponding definition in MIME, where it is an optional field used within the "message/external-body" content-type. In HTTP, it

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

SHOULD be sent whenever the message's length can be determined prior to being transferred, unless this is prohibited by the rules in section 4.4.

### 14.14 Content-Location

The Content-Location entity-header field MAY be used to supply the resource location for the entity enclosed in the message when that entity is accessible from a location separate from the requested resource's URI. A server SHOULD provide a Content-Location for the variant corresponding to the response entity; especially in the case where a resource has multiple entities associated with it, and those entities actually have separate locations by which they might be individually accessed, the server SHOULD provide a Content-Location for the particular variant which is returned.

```
    Content-Location = "Content-Location" ":"
                       ( absoluteURI | relativeURI )
```

The value of Content-Location also defines the base URI for the entity.

The Content-Location value is not a replacement for the original requested URI; it is only a statement of the location of the resource corresponding to this particular entity at the time of the request. Future requests MAY specify the Content-Location URI as the request-URI if the desire is to identify the source of that particular entity.

A cache cannot assume that an entity with a Content-Location different from the URI used to retrieve it can be used to respond to later requests on that Content-Location URI. However, the Content-Location can be used to differentiate between multiple entities retrieved from a single requested resource, as described in section 13.6.

If the Content-Location is a relative URI, the relative URI is interpreted relative to the Request-URI.

The meaning of the Content-Location header in PUT or POST requests is undefined; servers are free to ignore it in those cases.





### 14.15 Content-MD5

The Content-MD5 entity-header field, as defined in RFC 1864 [23], is an MD5 digest of the entity-body for the purpose of providing an end-to-end message integrity check (MIC) of the entity-body. (Note: a MIC is good for detecting accidental modification of the entity-body in transit, but is not proof against malicious attacks.)

```
     Content-MD5   = "Content-MD5" ":" md5-digest
     md5-digest   = <base64 of 128 bit MD5 digest as per RFC 1864>
```

The Content-MD5 header field MAY be generated by an origin server or client to function as an integrity check of the entity-body. Only origin servers or clients MAY generate the Content-MD5 header field;

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

proxies and gateways MUST NOT generate it, as this would defeat its value as an end-to-end integrity check. Any recipient of the entity-body, including gateways and proxies, MAY check that the digest value in this header field matches that of the entity-body as received.

The MD5 digest is computed based on the content of the entity-body, including any content-coding that has been applied, but not including any transfer-encoding applied to the message-body. If the message is received with a transfer-encoding, that encoding MUST be removed prior to checking the Content-MD5 value against the received entity.

This has the result that the digest is computed on the octets of the entity-body exactly as, and in the order that, they would be sent if no transfer-encoding were being applied.

HTTP extends RFC 1864 to permit the digest to be computed for MIME composite media-types (e.g., multipart/* and message/rfc822), but this does not change how the digest is computed as defined in the preceding paragraph.

There are several consequences of this. The entity-body for composite types MAY contain many body-parts, each with its own MIME and HTTP headers (including Content-MD5, Content-Transfer-Encoding, and Content-Encoding headers). If a body-part has a Content-Transfer-Encoding or Content-Encoding header, it is assumed that the content of the body-part has had the encoding applied, and the body-part is included in the Content-MD5 digest as is -- i.e., after the application. The Transfer-Encoding header field is not allowed within body-parts.

Conversion of all line breaks to CRLF MUST NOT be done before computing or checking the digest: the line break convention used in the text actually transmitted MUST be left unaltered when computing the digest.





Note: while the definition of Content-MD5 is exactly the same for HTTP as in RFC 1864 for MIME entity-bodies, there are several ways in which the application of Content-MD5 to HTTP entity-bodies differs from its application to MIME entity-bodies. One is that HTTP, unlike MIME, does not use Content-Transfer-Encoding, and does use Transfer-Encoding and Content-Encoding. Another is that HTTP more frequently uses binary content types than MIME, so it is worth noting that, in such cases, the byte order used to compute the digest is the transmission byte order defined for the type. Lastly, HTTP allows transmission of text types with any of several line break conventions and not just the canonical form using CRLF.

### 14.16 Content-Range

The Content-Range entity-header is sent with a partial entity-body to specify where in the full entity-body the partial body should be applied. Range units are defined in section 3.12.

```
Content-Range = "Content-Range" ":" content-range-spec

content-range-spec      = byte-content-range-spec
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

```
    byte-content-range-spec = bytes-unit SP
                              byte-range-resp-spec "/"
                              ( instance-length | "*" )

    byte-range-resp-spec = (first-byte-pos "-" last-byte-pos)
                                   | "*"
    instance-length           = 1*DIGIT
```

The header SHOULD indicate the total length of the full entity-body, unless this length is unknown or difficult to determine. The asterisk "*" character means that the instance-length is unknown at the time when the response was generated.

Unlike byte-ranges-specifier values (see section 14.35.1), a byte-range-resp-spec MUST only specify one range, and MUST contain absolute byte positions for both the first and last byte of the range.

A byte-content-range-spec with a byte-range-resp-spec whose last-byte-pos value is less than its first-byte-pos value, or whose instance-length value is less than or equal to its last-byte-pos value, is invalid. The recipient of an invalid byte-content-range-spec MUST ignore it and any content transferred along with it.

A server sending a response with status code 416 (Requested range not satisfiable) SHOULD include a Content-Range field with a byte-range-resp-spec of "*". The instance-length specifies the current length of





the selected resource. A response with status code 206 (Partial Content) MUST NOT include a Content-Range field with a byte-range-resp-spec of "*".

Examples of byte-content-range-spec values, assuming that the entity contains a total of 1234 bytes:

. The first 500 bytes:
bytes 0-499/1234

. The second 500 bytes:
bytes 500-999/1234

. All except for the first 500 bytes:
bytes 500-1233/1234

. The last 500 bytes:
bytes 734-1233/1234

When an HTTP message includes the content of a single range (for example, a response to a request for a single range, or to a request for a set of ranges that overlap without any holes), this content is transmitted with a Content-Range header, and a Content-Length header showing the number of bytes actually transferred. For example,

```
    HTTP/1.1 206 Partial content
    Date: Wed, 15 Nov 1995 06:25:24 GMT
    Last-Modified: Wed, 15 Nov 1995 04:58:08 GMT
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

```
    Content-Range: bytes 21010-47021/47022
    Content-Length: 26012
    Content-Type: image/gif
```

When an HTTP message includes the content of multiple ranges (for example, a response to a request for multiple non-overlapping ranges), these are transmitted as a multipart message. The multipart media type used for this purpose is "multipart/byteranges" as defined in appendix 19.2. See appendix 19.6.3 for a compatibility issue.

A response to a request for a single range MUST NOT be sent using the multipart/byteranges media type. A response to a request for multiple ranges, whose result is a single range, MAY be sent as a multipart/byteranges media type with one part. A client that cannot decode a multipart/byteranges message MUST NOT ask for multiple byte-ranges in a single request.

When a client requests multiple byte-ranges in one request, the server SHOULD return them in the order that they appeared in the request.





If the server ignores a byte-range-spec because it is syntactically invalid, the server SHOULD treat the request as if the invalid Range header field did not exist. (Normally, this means return a 200 response containing the full entity).

If the server receives a request (other than one including an If-Range request-header field) with an unsatisfiable Range request-header field (that is, all of whose byte-range-spec values have a first-byte-pos value greater than the current length of the selected resource), it SHOULD return a response code of 416 (Requested range not satisfiable) (section 10.4.17).

> Note: clients cannot depend on servers to send a 416 (Requested range not satisfiable) response instead of a 200 (OK) response for an unsatisfiable Range request-header, since not all servers implement this request-header.

### 14.17 Content-Type

The Content-Type entity-header field indicates the media type of the entity-body sent to the recipient or, in the case of the HEAD method, the media type that would have been sent had the request been a GET.

```
    Content-Type   = "Content-Type" ":" media-type
```

Media types are defined in section 3.7. An example of the field is

```
    Content-Type: text/html; charset=ISO-8859-4
```

Further discussion of methods for identifying the media type of an entity is provided in section 7.2.1.

### 14.18 Date

The Date general-header field represents the date and time at which

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

the message was originated, having the same semantics as orig-date in RFC 822. The field value is an HTTP-date, as described in section 3.3.1; it MUST be sent in RFC 1123 [8]-date format.

```
Date  = "Date" ":" HTTP-date
```

An example is

```
Date: Tue, 15 Nov 1994 08:12:31 GMT
```

Origin servers MUST include a Date header field in all responses, except in these cases:





1. If the response status code is 100 (Continue) or 101 (Switching Protocols), the response MAY include a Date header field, at the server's option.

2. If the response status code conveys a server error, e.g. 500 (Internal Server Error) or 503 (Service Unavailable), and it is inconvenient or impossible to generate a valid Date.

3. If the server does not have a clock that can provide a reasonable approximation of the current time, its responses MUST NOT include a Date header field. In this case, the rules in section 14.18.1 MUST be followed.

A received message that does not have a Date header field MUST be assigned one by the recipient if the message will be cached by that recipient or gatewayed via a protocol which requires a Date. An HTTP implementation without a clock MUST NOT cache responses without revalidating them on every use. An HTTP cache, especially a shared cache, SHOULD use a mechanism, such as NTP [28], to synchronize its clock with a reliable external standard.

Clients SHOULD only send a Date header field in messages that include an entity-body, as in the case of the PUT and POST requests, and even then it is optional. A client without a clock MUST NOT send a Date header field in a request.

The HTTP-date sent in a Date header SHOULD NOT represent a date and time subsequent to the generation of the message. It SHOULD represent the best available approximation of the date and time of message generation, unless the implementation has no means of generating a reasonably accurate date and time. In theory, the date ought to represent the moment just before the entity is generated. In practice, the date can be generated at any time during the message origination without affecting its semantic value.

### 14.18.1 Clockless Origin Server Operation

Some origin server implementations might not have a clock available. An origin server without a clock MUST NOT assign Expires or Last-Modified values to a response, unless these values were associated with the resource by a system or user with a reliable clock. It MAY assign an Expires value that is known, at or before server

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

configuration time, to be in the past (this allows "pre-expiration" of responses without storing separate Expires values for each resource).





### 14.19 ETag

The ETag response-header field provides the current value of the entity tag for the requested variant. The headers used with entity tags are described in sections 14.24, 14.26 and 14.44. The entity tag MAY be used for comparison with other entities from the same resource (see section 13.3.3).

```
ETag = "ETag" ":" entity-tag
```

Examples:

```
ETag: "xyzzy"
ETag: W/"xyzzy"
ETag: ""
```

### 14.20 Expect

The Expect request-header field is used to indicate that particular server behaviors are required by the client.

```
Expect       =  "Expect" ":" 1#expectation

expectation  =  "100-continue" | expectation-extension
expectation-extension =  token [ "=" ( token | quoted-string )
                         *expect-params ]
expect-params =  ";" token [ "=" ( token | quoted-string ) ]
```

A server that does not understand or is unable to comply with any of the expectation values in the Expect field of a request MUST respond with appropriate error status. The server MUST respond with a 417 (Expectation Failed) status if any of the expectations cannot be met or, if there are other problems with the request, some other 4xx status.

This header field is defined with extensible syntax to allow for future extensions. If a server receives a request containing an Expect field that includes an expectation-extension that it does not support, it MUST respond with a 417 (Expectation Failed) status.

Comparison of expectation values is case-insensitive for unquoted tokens (including the 100-continue token), and is case-sensitive for quoted-string expectation-extensions.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The Expect mechanism is hop-by-hop: that is, an HTTP/1.1 proxy MUST return a 417 (Expectation Failed) status if it receives a request with an expectation that it cannot meet. However, the Expect request-header itself is end-to-end; it MUST be forwarded if the request is forwarded.

Many older HTTP/1.0 and HTTP/1.1 applications do not understand the Expect header.

See section 8.2.3 for the use of the 100 (continue) status.

### 14.21 Expires

The Expires entity-header field gives the date/time after which the response is considered stale. A stale cache entry may not normally be returned by a cache (either a proxy cache or a user agent cache) unless it is first validated with the origin server (or with an intermediate cache that has a fresh copy of the entity). See section 13.2 for further discussion of the expiration model.

The presence of an Expires field does not imply that the original resource will change or cease to exist at, before, or after that time.

The format is an absolute date and time as defined by HTTP-date in section 3.3.1; it MUST be in RFC 1123 date format:

```
Expires = "Expires" ":" HTTP-date
```

An example of its use is

```
Expires: Thu, 01 Dec 1994 16:00:00 GMT
```

Note: if a response includes a Cache-Control field with the max-age directive (see section 14.9.3), that directive overrides the Expires field.

HTTP/1.1 clients and caches MUST treat other invalid date formats, especially including the value "0", as in the past (i.e., "already expired").

To mark a response as "already expired," an origin server sends an Expires date that is equal to the Date header value. (See the rules for expiration calculations in section 13.2.4.)



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

To mark a response as "never expires," an origin server sends an Expires date approximately one year from the time the response is sent. HTTP/1.1 servers SHOULD NOT send Expires dates more than one year in the future.

The presence of an Expires header field with a date value of some time in the future on a response that otherwise would by default be non-cacheable indicates that the response is cacheable, unless indicated otherwise by a Cache-Control header field (section 14.9).

### 14.22 From

The From request-header field, if given, SHOULD contain an Internet e-mail address for the human user who controls the requesting user agent. The address SHOULD be machine-usable, as defined by "mailbox" in RFC 822 [9] as updated by RFC 1123 [8]:

```
    From   = "From" ":" mailbox
```

An example is:

```
    From: webmaster@w3.org
```

This header field MAY be used for logging purposes and as a means for identifying the source of invalid or unwanted requests. It SHOULD NOT be used as an insecure form of access protection. The interpretation of this field is that the request is being performed on behalf of the person given, who accepts responsibility for the method performed. In particular, robot agents SHOULD include this header so that the person responsible for running the robot can be contacted if problems occur on the receiving end.

The Internet e-mail address in this field MAY be separate from the Internet host which issued the request. For example, when a request is passed through a proxy the original issuer's address SHOULD be used.

The client SHOULD NOT send the From header field without the user's approval, as it might conflict with the user's privacy interests or their site's security policy. It is strongly recommended that the user be able to disable, enable, and modify the value of this field at any time prior to a request.

### 14.23 Host

The Host request-header field specifies the Internet host and port number of the resource being requested, as obtained from the original URI given by the user or referring resource (generally an HTTP URL,





as described in section 3.2.2). The Host field value MUST represent the naming authority of the origin server or gateway given by the original URL. This allows the origin server or gateway to differentiate between internally-ambiguous URLs, such as the root "/" URL of a server for multiple host names on a single IP address.

```
    Host = "Host" ":" host [ ":" port ] ; Section 3.2.2
```

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1191
Keysight, EX-1045
Page 1191

A "host" without any trailing port information implies the default port for the service requested (e.g., "80" for an HTTP URL). For example, a request on the origin server for <http://www.w3.org/pub/WWW/> would properly include:

```
GET /pub/WWW/ HTTP/1.1
Host: www.w3.org
```

A client MUST include a Host header field in all HTTP/1.1 request messages . If the requested URI does not include an Internet host name for the service being requested, then the Host header field MUST be given with an empty value. An HTTP/1.1 proxy MUST ensure that any request message it forwards does contain an appropriate Host header field that identifies the service being requested by the proxy. All Internet-based HTTP/1.1 servers MUST respond with a 400 (Bad Request) status code to any HTTP/1.1 request message which lacks a Host header field.

See sections 5.2 and 19.6.1.1 for other requirements relating to Host.

### 14.24 If-Match

The If-Match request-header field is used with a method to make it conditional. A client that has one or more entities previously obtained from the resource can verify that one of those entities is current by including a list of their associated entity tags in the If-Match header field. Entity tags are defined in section 3.11. The purpose of this feature is to allow efficient updates of cached information with a minimum amount of transaction overhead. It is also used, on updating requests, to prevent inadvertent modification of the wrong version of a resource. As a special case, the value "*" matches any current entity of the resource.

```
If-Match = "If-Match" ":" ( "*" | 1#entity-tag )
```

If any of the entity tags match the entity tag of the entity that would have been returned in the response to a similar GET request (without the If-Match header) on that resource, or if "*" is given





and any current entity exists for that resource, then the server MAY perform the requested method as if the If-Match header field did not exist.

A server MUST use the strong comparison function (see section 13.3.3) to compare the entity tags in If-Match.

If none of the entity tags match, or if "*" is given and no current entity exists, the server MUST NOT perform the requested method, and MUST return a 412 (Precondition Failed) response. This behavior is most useful when the client wants to prevent an updating method, such as PUT, from modifying a resource that has changed since the client last retrieved it.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1192
Keysight, EX-1045
Page 1192

If the request would, without the If-Match header field, result in anything other than a 2xx or 412 status, then the If-Match header MUST be ignored.

The meaning of "If-Match: *" is that the method SHOULD be performed if the representation selected by the origin server (or by a cache, possibly using the Vary mechanism, see section 14.44) exists, and MUST NOT be performed if the representation does not exist.

A request intended to update a resource (e.g., a PUT) MAY include an If-Match header field to signal that the request method MUST NOT be applied if the entity corresponding to the If-Match value (a single entity tag) is no longer a representation of that resource. This allows the user to indicate that they do not wish the request to be successful if the resource has been changed without their knowledge. Examples:

```
    If-Match: "xyzzy"
    If-Match: "xyzzy", "r2d2xxxx", "c3piozzzz"
    If-Match: *
```

The result of a request having both an If-Match header field and either an If-None-Match or an If-Modified-Since header fields is undefined by this specification.

### 14.25 If-Modified-Since

The If-Modified-Since request-header field is used with a method to make it conditional: if the requested variant has not been modified since the time specified in this field, an entity will not be returned from the server; instead, a 304 (not modified) response will be returned without any message-body.

```
    If-Modified-Since = "If-Modified-Since" ":" HTTP-date
```





An example of the field is:

```
    If-Modified-Since: Sat, 29 Oct 1994 19:43:31 GMT
```

A GET method with an If-Modified-Since header and no Range header requests that the identified entity be transferred only if it has been modified since the date given by the If-Modified-Since header. The algorithm for determining this includes the following cases:

a) If the request would normally result in anything other than a 200 (OK) status, or if the passed If-Modified-Since date is invalid, the response is exactly the same as for a normal GET. A date which is later than the server's current time is invalid.

b) If the variant has been modified since the If-Modified-Since date, the response is exactly the same as for a normal GET.

c) If the variant has not been modified since a valid If-Modified-Since date, the server SHOULD return a 304 (Not Modified) response.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The purpose of this feature is to allow efficient updates of cached information with a minimum amount of transaction overhead.

Note: The Range request-header field modifies the meaning of If-Modified-Since; see section 14.35 for full details.

Note: If-Modified-Since times are interpreted by the server, whose clock might not be synchronized with the client.

Note: When handling an If-Modified-Since header field, some servers will use an exact date comparison function, rather than a less-than function, for deciding whether to send a 304 (Not Modified) response. To get best results when sending an If-Modified-Since header field for cache validation, clients are advised to use the exact date string received in a previous Last-Modified header field whenever possible.

Note: If a client uses an arbitrary date in the If-Modified-Since header instead of a date taken from the Last-Modified header for the same request, the client should be aware of the fact that this date is interpreted in the server's understanding of time. The client should consider unsynchronized clocks and rounding problems due to the different encodings of time between the client and server. This includes the possibility of race conditions if the document has changed between the time it was first requested and the If-Modified-Since date of a subsequent request, and the





possibility of clock-skew-related problems if the If-Modified-Since date is derived from the client's clock without correction to the server's clock. Corrections for different time bases between client and server are at best approximate due to network latency.

The result of a request having both an If-Modified-Since header field and either an If-Match or an If-Unmodified-Since header fields is undefined by this specification.

### 14.26 If-None-Match

The If-None-Match request-header field is used with a method to make it conditional. A client that has one or more entities previously obtained from the resource can verify that none of those entities is current by including a list of their associated entity tags in the If-None-Match header field. The purpose of this feature is to allow efficient updates of cached information with a minimum amount of transaction overhead. It is also used to prevent a method (e.g. PUT) from inadvertently modifying an existing resource when the client believes that the resource does not exist.

As a special case, the value "*" matches any current entity of the resource.

```
If-None-Match = "If-None-Match" ":" ( "*" | 1#entity-tag )
```

If any of the entity tags match the entity tag of the entity that

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

would have been returned in the response to a similar GET request (without the If-None-Match header) on that resource, or if "*" is given and any current entity exists for that resource, then the server MUST NOT perform the requested method, unless required to do so because the resource's modification date fails to match that supplied in an If-Modified-Since header field in the request. Instead, if the request method was GET or HEAD, the server SHOULD respond with a 304 (Not Modified) response, including the cache-related header fields (particularly ETag) of one of the entities that matched. For all other request methods, the server MUST respond with a status of 412 (Precondition Failed).

See section 13.3.3 for rules on how to determine if two entities tags match. The weak comparison function can only be used with GET or HEAD requests.





If none of the entity tags match, then the server MAY perform the requested method as if the If-None-Match header field did not exist, but MUST also ignore any If-Modified-Since header field(s) in the request. That is, if no entity tags match, then the server MUST NOT return a 304 (Not Modified) response.

If the request would, without the If-None-Match header field, result in anything other than a 2xx or 304 status, then the If-None-Match header MUST be ignored. (See section 13.3.4 for a discussion of server behavior when both If-Modified-Since and If-None-Match appear in the same request.)

The meaning of "If-None-Match: *" is that the method MUST NOT be performed if the representation selected by the origin server (or by a cache, possibly using the Vary mechanism, see section 14.44) exists, and SHOULD be performed if the representation does not exist. This feature is intended to be useful in preventing races between PUT operations.

Examples:

```
    If-None-Match: "xyzzy"
    If-None-Match: W/"xyzzy"
    If-None-Match: "xyzzy", "r2d2xxxx", "c3piozzzz"
    If-None-Match: W/"xyzzy", W/"r2d2xxxx", W/"c3piozzzz"
    If-None-Match: *
```

The result of a request having both an If-None-Match header field and either an If-Match or an If-Unmodified-Since header fields is undefined by this specification.

### 14.27 If-Range

If a client has a partial copy of an entity in its cache, and wishes to have an up-to-date copy of the entire entity in its cache, it

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

could use the Range request-header with a conditional GET (using either or both of If-Unmodified-Since and If-Match.) However, if the condition fails because the entity has been modified, the client would then have to make a second request to obtain the entire current entity-body.

The If-Range header allows a client to "short-circuit" the second request. Informally, its meaning is `if the entity is unchanged, send me the part(s) that I am missing; otherwise, send me the entire new entity'.

```
If-Range = "If-Range" ":" ( entity-tag | HTTP-date )
```





If the client has no entity tag for an entity, but does have a Last-Modified date, it MAY use that date in an If-Range header. (The server can distinguish between a valid HTTP-date and any form of entity-tag by examining no more than two characters.) The If-Range header SHOULD only be used together with a Range header, and MUST be ignored if the request does not include a Range header, or if the server does not support the sub-range operation.

If the entity tag given in the If-Range header matches the current entity tag for the entity, then the server SHOULD provide the specified sub-range of the entity using a 206 (Partial content) response. If the entity tag does not match, then the server SHOULD return the entire entity using a 200 (OK) response.

### 14.28 If-Unmodified-Since

The If-Unmodified-Since request-header field is used with a method to make it conditional. If the requested resource has not been modified since the time specified in this field, the server SHOULD perform the requested operation as if the If-Unmodified-Since header were not present.

If the requested variant has been modified since the specified time, the server MUST NOT perform the requested operation, and MUST return a 412 (Precondition Failed).

```
If-Unmodified-Since = "If-Unmodified-Since" ":" HTTP-date
```

An example of the field is:

```
If-Unmodified-Since: Sat, 29 Oct 1994 19:43:31 GMT
```

If the request normally (i.e., without the If-Unmodified-Since header) would result in anything other than a 2xx or 412 status, the If-Unmodified-Since header SHOULD be ignored.

If the specified date is invalid, the header is ignored.

The result of a request having both an If-Unmodified-Since header field and either an If-None-Match or an If-Modified-Since header fields is undefined by this specification.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

14.29 Last-Modified

The Last-Modified entity-header field indicates the date and time at which the origin server believes the variant was last modified.

```
Last-Modified  = "Last-Modified" ":" HTTP-date
```





An example of its use is

```
Last-Modified: Tue, 15 Nov 1994 12:45:26 GMT
```

The exact meaning of this header field depends on the implementation of the origin server and the nature of the original resource. For files, it may be just the file system last-modified time. For entities with dynamically included parts, it may be the most recent of the set of last-modify times for its component parts. For database gateways, it may be the last-update time stamp of the record. For virtual objects, it may be the last time the internal state changed.

An origin server MUST NOT send a Last-Modified date which is later than the server's time of message origination. In such cases, where the resource's last modification would indicate some time in the future, the server MUST replace that date with the message origination date.

An origin server SHOULD obtain the Last-Modified value of the entity as close as possible to the time that it generates the Date value of its response. This allows a recipient to make an accurate assessment of the entity's modification time, especially if the entity changes near the time that the response is generated.

HTTP/1.1 servers SHOULD send Last-Modified whenever feasible.

14.30 Location

The Location response-header field is used to redirect the recipient to a location other than the Request-URI for completion of the request or identification of a new resource. For 201 (Created) responses, the Location is that of the new resource which was created by the request. For 3xx responses, the location SHOULD indicate the server's preferred URI for automatic redirection to the resource. The field value consists of a single absolute URI.

```
Location       = "Location" ":" absoluteURI
```

An example is:

```
Location: http://www.w3.org/pub/WWW/People.html
```

Note: The Content-Location header field (section 14.14) differs from Location in that the Content-Location identifies the original location of the entity enclosed in the request. It is therefore possible for a response to contain header fields for both Location and Content-Location. Also see section 13.10 for cache requirements of some methods.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

### 14.31 Max-Forwards

The Max-Forwards request-header field provides a mechanism with the TRACE (section 9.8) and OPTIONS (section 9.2) methods to limit the number of proxies or gateways that can forward the request to the next inbound server. This can be useful when the client is attempting to trace a request chain which appears to be failing or looping in mid-chain.

```
Max-Forwards   = "Max-Forwards" ":" 1*DIGIT
```

The Max-Forwards value is a decimal integer indicating the remaining number of times this request message may be forwarded.

Each proxy or gateway recipient of a TRACE or OPTIONS request containing a Max-Forwards header field MUST check and update its value prior to forwarding the request. If the received value is zero (0), the recipient MUST NOT forward the request; instead, it MUST respond as the final recipient. If the received Max-Forwards value is greater than zero, then the forwarded message MUST contain an updated Max-Forwards field with a value decremented by one (1).

The Max-Forwards header field MAY be ignored for all other methods defined by this specification and for any extension methods for which it is not explicitly referred to as part of that method definition.

### 14.32 Pragma

The Pragma general-header field is used to include implementation-specific directives that might apply to any recipient along the request/response chain. All pragma directives specify optional behavior from the viewpoint of the protocol; however, some systems MAY require that behavior be consistent with the directives.

```
Pragma            = "Pragma" ":" 1#pragma-directive
pragma-directive  = "no-cache" | extension-pragma
extension-pragma  = token [ "=" ( token | quoted-string ) ]
```

When the no-cache directive is present in a request message, an application SHOULD forward the request toward the origin server even if it has a cached copy of what is being requested. This pragma directive has the same semantics as the no-cache cache-directive (see section 14.9) and is defined here for backward compatibility with HTTP/1.0. Clients SHOULD include both header fields when a no-cache request is sent to a server not known to be HTTP/1.1 compliant.



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

Pragma directives MUST be passed through by a proxy or gateway application, regardless of their significance to that application, since the directives might be applicable to all recipients along the request/response chain. It is not possible to specify a pragma for a specific recipient; however, any pragma directive not relevant to a recipient SHOULD be ignored by that recipient.

HTTP/1.1 caches SHOULD treat "Pragma: no-cache" as if the client had sent "Cache-Control: no-cache". No new Pragma directives will be defined in HTTP.

Note: because the meaning of "Pragma: no-cache as a response header field is not actually specified, it does not provide a reliable replacement for "Cache-Control: no-cache" in a response

14.33 Proxy-Authenticate

The Proxy-Authenticate response-header field MUST be included as part of a 407 (Proxy Authentication Required) response. The field value consists of a challenge that indicates the authentication scheme and parameters applicable to the proxy for this Request-URI.

Proxy-Authenticate = "Proxy-Authenticate" ":" 1#challenge

The HTTP access authentication process is described in "HTTP Authentication: Basic and Digest Access Authentication" [43]. Unlike WWW-Authenticate, the Proxy-Authenticate header field applies only to the current connection and SHOULD NOT be passed on to downstream clients. However, an intermediate proxy might need to obtain its own credentials by requesting them from the downstream client, which in some circumstances will appear as if the proxy is forwarding the Proxy-Authenticate header field.

14.34 Proxy-Authorization

The Proxy-Authorization request-header field allows the client to identify itself (or its user) to a proxy which requires authentication. The Proxy-Authorization field value consists of credentials containing the authentication information of the user agent for the proxy and/or realm of the resource being requested.

Proxy-Authorization = "Proxy-Authorization" ":" credentials

The HTTP access authentication process is described in "HTTP Authentication: Basic and Digest Access Authentication" [43] . Unlike Authorization, the Proxy-Authorization header field applies only to the next outbound proxy that demanded authentication using the Proxy-Authenticate field. When multiple proxies are used in a chain, the





Proxy-Authorization header field is consumed by the first outbound proxy that was expecting to receive credentials. A proxy MAY relay the credentials from the client request to the next proxy if that is the mechanism by which the proxies cooperatively authenticate a given request.

14.35 Range

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1199
Keysight, EX-1045
Page 1199

### 14.35.1 Byte Ranges

Since all HTTP entities are represented in HTTP messages as sequences of bytes, the concept of a byte range is meaningful for any HTTP entity. (However, not all clients and servers need to support byte-range operations.)

Byte range specifications in HTTP apply to the sequence of bytes in the entity-body (not necessarily the same as the message-body).

A byte range operation MAY specify a single range of bytes, or a set of ranges within a single entity.

```
ranges-specifier = byte-ranges-specifier
byte-ranges-specifier = bytes-unit "=" byte-range-set
byte-range-set  = 1#( byte-range-spec | suffix-byte-range-spec )
byte-range-spec = first-byte-pos "-" [last-byte-pos]
first-byte-pos  = 1*DIGIT
last-byte-pos   = 1*DIGIT
```

The first-byte-pos value in a byte-range-spec gives the byte-offset of the first byte in a range. The last-byte-pos value gives the byte-offset of the last byte in the range; that is, the byte positions specified are inclusive. Byte offsets start at zero.

If the last-byte-pos value is present, it MUST be greater than or equal to the first-byte-pos in that byte-range-spec, or the byte-range-spec is syntactically invalid. The recipient of a byte-range-set that includes one or more syntactically invalid byte-range-spec values MUST ignore the header field that includes that byte-range-set.

If the last-byte-pos value is absent, or if the value is greater than or equal to the current length of the entity-body, last-byte-pos is taken to be equal to one less than the current length of the entity-body in bytes.

By its choice of last-byte-pos, a client can limit the number of bytes retrieved without knowing the size of the entity.





```
suffix-byte-range-spec = "-" suffix-length
suffix-length = 1*DIGIT
```

A suffix-byte-range-spec is used to specify the suffix of the entity-body, of a length given by the suffix-length value. (That is, this form specifies the last N bytes of an entity-body.) If the entity is shorter than the specified suffix-length, the entire entity-body is used.

If a syntactically valid byte-range-set includes at least one byte-range-spec whose first-byte-pos is less than the current length of the entity-body, or at least one suffix-byte-range-spec with a non-zero suffix-length, then the byte-range-set is satisfiable. Otherwise, the byte-range-set is unsatisfiable. If the byte-range-set

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

is unsatisfiable, the server SHOULD return a response with a status of 416 (Requested range not satisfiable). Otherwise, the server SHOULD return a response with a status of 206 (Partial Content) containing the satisfiable ranges of the entity-body.

Examples of byte-ranges-specifier values (assuming an entity-body of length 10000):

- The first 500 bytes (byte offsets 0-499, inclusive): bytes=0-499
- The second 500 bytes (byte offsets 500-999, inclusive): bytes=500-999
- The final 500 bytes (byte offsets 9500-9999, inclusive): bytes=-500
- Or bytes=9500-
- The first and last bytes only (bytes 0 and 9999): bytes=0-0,-1
- Several legal but not canonical specifications of the second 500 bytes (byte offsets 500-999, inclusive):
  bytes=500-600,601-999
  bytes=500-700,601-999

14.35.2 Range Retrieval Requests

HTTP retrieval requests using conditional or unconditional GET methods MAY request one or more sub-ranges of the entity, instead of the entire entity, using the Range request header, which applies to the entity returned as the result of the request:

Range = "Range" ":" ranges-specifier





A server MAY ignore the Range header. However, HTTP/1.1 origin servers and intermediate caches ought to support byte ranges when possible, since Range supports efficient recovery from partially failed transfers, and supports efficient partial retrieval of large entities.

If the server supports the Range header and the specified range or ranges are appropriate for the entity:

- The presence of a Range header in an unconditional GET modifies what is returned if the GET is otherwise successful. In other words, the response carries a status code of 206 (Partial Content) instead of 200 (OK).
- The presence of a Range header in a conditional GET (a request using one or both of If-Modified-Since and If-None-Match, or one or both of If-Unmodified-Since and If-Match) modifies what is returned if the GET is otherwise successful and the condition is true. It does not affect the 304 (Not Modified) response returned if the conditional is false.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1201
Keysight, EX-1045
Page 1201

In some cases, it might be more appropriate to use the If-Range header (see section 14.27) in addition to the Range header.

If a proxy that supports ranges receives a Range request, forwards the request to an inbound server, and receives an entire entity in reply, it SHOULD only return the requested range to its client. It SHOULD store the entire received response in its cache if that is consistent with its cache allocation policies.

14.36 Referer

The Referer[sic] request-header field allows the client to specify, for the server's benefit, the address (URI) of the resource from which the Request-URI was obtained (the "referrer", although the header field is misspelled.) The Referer request-header allows a server to generate lists of back-links to resources for interest, logging, optimized caching, etc. It also allows obsolete or mistyped links to be traced for maintenance. The Referer field MUST NOT be sent if the Request-URI was obtained from a source that does not have its own URI, such as input from the user keyboard.

```
    Referer        = "Referer" ":" ( absoluteURI | relativeURI )
```

Example:

```
    Referer: http://www.w3.org/hypertext/DataSources/Overview.html
```





If the field value is a relative URI, it SHOULD be interpreted relative to the Request-URI. The URI MUST NOT include a fragment. See section 15.1.3 for security considerations.

14.37 Retry-After

The Retry-After response-header field can be used with a 503 (Service Unavailable) response to indicate how long the service is expected to be unavailable to the requesting client. This field MAY also be used with any 3xx (Redirection) response to indicate the minimum time the user-agent is asked wait before issuing the redirected request. The value of this field can be either an HTTP-date or an integer number of seconds (in decimal) after the time of the response.

```
    Retry-After  = "Retry-After" ":" ( HTTP-date | delta-seconds )
```

Two examples of its use are

```
    Retry-After: Fri, 31 Dec 1999 23:59:59 GMT
    Retry-After: 120
```

In the latter example, the delay is 2 minutes.

14.38 Server

The Server response-header field contains information about the software used by the origin server to handle the request. The field can contain multiple product tokens (section 3.8) and comments

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

identifying the server and any significant subproducts. The product tokens are listed in order of their significance for identifying the application.

```
    Server         = "Server" ":" 1*( product | comment )
```

Example:

```
    Server: CERN/3.0 libwww/2.17
```

If the response is being forwarded through a proxy, the proxy application MUST NOT modify the Server response-header. Instead, it SHOULD include a Via field (as described in section 14.45).

> Note: Revealing the specific software version of the server might allow the server machine to become more vulnerable to attacks against software that is known to contain security holes. Server implementors are encouraged to make this field a configurable option.





## 14.39 TE

The TE request-header field indicates what extension transfer-codings it is willing to accept in the response and whether or not it is willing to accept trailer fields in a chunked transfer-coding. Its value may consist of the keyword "trailers" and/or a comma-separated list of extension transfer-coding names with optional accept parameters (as described in section 3.6).

```
    TE        = "TE" ":" #( t-codings )
    t-codings = "trailers" | ( transfer-extension [ accept-params ] )
```

The presence of the keyword "trailers" indicates that the client is willing to accept trailer fields in a chunked transfer-coding, as defined in section 3.6.1. This keyword is reserved for use with transfer-coding values even though it does not itself represent a transfer-coding.

Examples of its use are:

```
    TE: deflate
    TE:
    TE: trailers, deflate;q=0.5
```

The TE header field only applies to the immediate connection. Therefore, the keyword MUST be supplied within a Connection header field (section 14.10) whenever TE is present in an HTTP/1.1 message.

A server tests whether a transfer-coding is acceptable, according to a TE field, using these rules:

1. The "chunked" transfer-coding is always acceptable. If the keyword "trailers" is listed, the client indicates that it is willing to accept trailer fields in the chunked response on behalf of itself and any downstream clients. The implication is

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

that, if given, the client is stating that either all downstream clients are willing to accept trailer fields in the forwarded response, or that it will attempt to buffer the response on behalf of downstream recipients.

Note: HTTP/1.1 does not define any means to limit the size of a chunked response such that a client can be assured of buffering the entire response.

2. If the transfer-coding being tested is one of the transfer-codings listed in the TE field, then it is acceptable unless it is accompanied by a qvalue of 0. (As defined in section 3.9, a qvalue of 0 means "not acceptable.")





3. If multiple transfer-codings are acceptable, then the acceptable transfer-coding with the highest non-zero qvalue is preferred. The "chunked" transfer-coding always has a qvalue of 1.

If the TE field-value is empty or if no TE field is present, the only transfer-coding is "chunked". A message with no transfer-coding is always acceptable.

### 14.40 Trailer

The Trailer general field value indicates that the given set of header fields is present in the trailer of a message encoded with chunked transfer-coding.

```
Trailer  = "Trailer" ":" 1#field-name
```

An HTTP/1.1 message SHOULD include a Trailer header field in a message using chunked transfer-coding with a non-empty trailer. Doing so allows the recipient to know which header fields to expect in the trailer.

If no Trailer header field is present, the trailer SHOULD NOT include any header fields. See section 3.6.1 for restrictions on the use of trailer fields in a "chunked" transfer-coding.

Message header fields listed in the Trailer header field MUST NOT include the following header fields:

- Transfer-Encoding
- Content-Length
- Trailer

### 14.41 Transfer-Encoding

The Transfer-Encoding general-header field indicates what (if any) type of transformation has been applied to the message body in order to safely transfer it between the sender and the recipient. This differs from the content-coding in that the transfer-coding is a property of the message, not of the entity.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

```
     Transfer-Encoding       = "Transfer-Encoding" ":" 1#transfer-coding
```

Transfer-codings are defined in section 3.6. An example is:

```
     Transfer-Encoding: chunked
```





If multiple encodings have been applied to an entity, the transfer-codings MUST be listed in the order in which they were applied. Additional information about the encoding parameters MAY be provided by other entity-header fields not defined by this specification.

Many older HTTP/1.0 applications do not understand the Transfer-Encoding header.

### 14.42 Upgrade

The Upgrade general-header allows the client to specify what additional communication protocols it supports and would like to use if the server finds it appropriate to switch protocols. The server MUST use the Upgrade header field within a 101 (Switching Protocols) response to indicate which protocol(s) are being switched.

```
    Upgrade        = "Upgrade" ":" 1#product
```

For example,

```
    Upgrade: HTTP/2.0, SHTTP/1.3, IRC/6.9, RTA/x11
```

The Upgrade header field is intended to provide a simple mechanism for transition from HTTP/1.1 to some other, incompatible protocol. It does so by allowing the client to advertise its desire to use another protocol, such as a later version of HTTP with a higher major version number, even though the current request has been made using HTTP/1.1. This eases the difficult transition between incompatible protocols by allowing the client to initiate a request in the more commonly supported protocol while indicating to the server that it would like to use a "better" protocol if available (where "better" is determined by the server, possibly according to the nature of the method and/or resource being requested).

The Upgrade header field only applies to switching application-layer protocols upon the existing transport-layer connection. Upgrade cannot be used to insist on a protocol change; its acceptance and use by the server is optional. The capabilities and nature of the application-layer communication after the protocol change is entirely dependent upon the new protocol chosen, although the first action after changing the protocol MUST be a response to the initial HTTP request containing the Upgrade header field.

The Upgrade header field only applies to the immediate connection. Therefore, the upgrade keyword MUST be supplied within a Connection header field (section 14.10) whenever Upgrade is present in an HTTP/1.1 message.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The Upgrade header field cannot be used to indicate a switch to a protocol on a different connection. For that purpose, it is more appropriate to use a 301, 302, 303, or 305 redirection response.

This specification only defines the protocol name "HTTP" for use by the family of Hypertext Transfer Protocols, as defined by the HTTP version rules of section 3.1 and future updates to this specification. Any token can be used as a protocol name; however, it will only be useful if both the client and server associate the name with the same protocol.

### 14.43 User-Agent

The User-Agent request-header field contains information about the user agent originating the request. This is for statistical purposes, the tracing of protocol violations, and automated recognition of user agents for the sake of tailoring responses to avoid particular user agent limitations. User agents SHOULD include this field with requests. The field can contain multiple product tokens (section 3.8) and comments identifying the agent and any subproducts which form a significant part of the user agent. By convention, the product tokens are listed in order of their significance for identifying the application.

```
User-Agent     = "User-Agent" ":" 1*( product | comment )
```

Example:

```
User-Agent: CERN-LineMode/2.15 libwww/2.17b3
```

### 14.44 Vary

The Vary field value indicates the set of request-header fields that fully determines, while the response is fresh, whether a cache is permitted to use the response to reply to a subsequent request without revalidation. For uncacheable or stale responses, the Vary field value advises the user agent about the criteria that were used to select the representation. A Vary field value of "*" implies that a cache cannot determine from the request headers of a subsequent request whether this response is the appropriate representation. See section 13.6 for use of the Vary header field by caches.

```
Vary  = "Vary" ":" ( "*" | 1#field-name )
```

An HTTP/1.1 server SHOULD include a Vary header field with any cacheable response that is subject to server-driven negotiation. Doing so allows a cache to properly interpret future requests on that resource and informs the user agent about the presence of negotiation



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1206
Keysight, EX-1045
Page 1206

on that resource. A server MAY include a Vary header field with a non-cacheable response that is subject to server-driven negotiation, since this might provide the user agent with useful information about the dimensions over which the response varies at the time of the response.

A Vary field value consisting of a list of field-names signals that the representation selected for the response is based on a selection algorithm which considers ONLY the listed request-header field values in selecting the most appropriate representation. A cache MAY assume that the same selection will be made for future requests with the same values for the listed field names, for the duration of time for which the response is fresh.

The field-names given are not limited to the set of standard request-header fields defined by this specification. Field names are case-insensitive.

A Vary field value of "*" signals that unspecified parameters not limited to the request-headers (e.g., the network address of the client), play a role in the selection of the response representation. The "*" value MUST NOT be generated by a proxy server; it may only be generated by an origin server.

### 14.45 Via

The Via general-header field MUST be used by gateways and proxies to indicate the intermediate protocols and recipients between the user agent and the server on requests, and between the origin server and the client on responses. It is analogous to the "Received" field of RFC 822 [9] and is intended to be used for tracking message forwards, avoiding request loops, and identifying the protocol capabilities of all senders along the request/response chain.

```
Via =  "Via" ":" 1#( received-protocol received-by [ comment ] )
received-protocol = [ protocol-name "/" ] protocol-version
protocol-name     = token
protocol-version  = token
received-by       = ( host [ ":" port ] ) | pseudonym
pseudonym         = token
```

The received-protocol indicates the protocol version of the message received by the server or client along each segment of the request/response chain. The received-protocol version is appended to the Via field value when the message is forwarded so that information about the protocol capabilities of upstream applications remains visible to all recipients.





The protocol-name is optional if and only if it would be "HTTP". The received-by field is normally the host and optional port number of a recipient server or client that subsequently forwarded the message. However, if the real host is considered to be sensitive information, it MAY be replaced by a pseudonym. If the port is not given, it MAY be assumed to be the default port of the received-protocol.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1207
Keysight, EX-1045
Page 1207

Multiple Via field values represents each proxy or gateway that has forwarded the message. Each recipient MUST append its information such that the end result is ordered according to the sequence of forwarding applications.

Comments MAY be used in the Via header field to identify the software of the recipient proxy or gateway, analogous to the User-Agent and Server header fields. However, all comments in the Via field are optional and MAY be removed by any recipient prior to forwarding the message.

For example, a request message could be sent from an HTTP/1.0 user agent to an internal proxy code-named "fred", which uses HTTP/1.1 to forward the request to a public proxy at nowhere.com, which completes the request by forwarding it to the origin server at www.ics.uci.edu. The request received by www.ics.uci.edu would then have the following Via header field:

```
Via: 1.0 fred, 1.1 nowhere.com (Apache/1.1)
```

Proxies and gateways used as a portal through a network firewall SHOULD NOT, by default, forward the names and ports of hosts within the firewall region. This information SHOULD only be propagated if explicitly enabled. If not enabled, the received-by host of any host behind the firewall SHOULD be replaced by an appropriate pseudonym for that host.

For organizations that have strong privacy requirements for hiding internal structures, a proxy MAY combine an ordered subsequence of Via header field entries with identical received-protocol values into a single such entry. For example,

```
Via: 1.0 ricky, 1.1 ethel, 1.1 fred, 1.0 lucy
```

could be collapsed to

```
Via: 1.0 ricky, 1.1 mertz, 1.0 lucy
```





Applications SHOULD NOT combine multiple entries unless they are all under the same organizational control and the hosts have already been replaced by pseudonyms. Applications MUST NOT combine entries which have different received-protocol values.

### 14.46 Warning

The Warning general-header field is used to carry additional information about the status or transformation of a message which might not be reflected in the message. This information is typically used to warn about a possible lack of semantic transparency from caching operations or transformations applied to the entity body of the message.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

Warning headers are sent with responses using:

```
    Warning    = "Warning" ":" 1#warning-value

    warning-value = warn-code SP warn-agent SP warn-text
                                          [SP warn-date]

    warn-code  = 3DIGIT
    warn-agent = ( host [ ":" port ] ) | pseudonym
                    ; the name or pseudonym of the server adding
                    ; the Warning header, for use in debugging
    warn-text  = quoted-string
    warn-date  = <"> HTTP-date <">
```

A response MAY carry more than one Warning header.

The warn-text SHOULD be in a natural language and character set that is most likely to be intelligible to the human user receiving the response. This decision MAY be based on any available knowledge, such as the location of the cache or user, the Accept-Language field in a request, the Content-Language field in a response, etc. The default language is English and the default character set is ISO-8859-1.

If a character set other than ISO-8859-1 is used, it MUST be encoded in the warn-text using the method described in RFC 2047 [14].

Warning headers can in general be applied to any message, however some specific warn-codes are specific to caches and can only be applied to response messages. New Warning headers SHOULD be added after any existing Warning headers. A cache MUST NOT delete any Warning header that it received with a message. However, if a cache successfully validates a cache entry, it SHOULD remove any Warning headers previously attached to that entry except as specified for





specific Warning codes. It MUST then add any Warning headers received in the validating response. In other words, Warning headers are those that would be attached to the most recent relevant response.

When multiple Warning headers are attached to a response, the user agent ought to inform the user of as many of them as possible, in the order that they appear in the response. If it is not possible to inform the user of all of the warnings, the user agent SHOULD follow these heuristics:

- Warnings that appear early in the response take priority over those appearing later in the response.

- Warnings in the user's preferred character set take priority over warnings in other character sets but with identical warn-codes and warn-agents.

Systems that generate multiple Warning headers SHOULD order them with this user agent behavior in mind.

Requirements for the behavior of caches with respect to Warnings are

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

stated in section 13.1.2.

This is a list of the currently-defined warn-codes, each with a recommended warn-text in English, and a description of its meaning.

110 Response is stale
MUST be included whenever the returned response is stale.

111 Revalidation failed
MUST be included if a cache returns a stale response because an attempt to revalidate the response failed, due to an inability to reach the server.

112 Disconnected operation
SHOULD be included if the cache is intentionally disconnected from the rest of the network for a period of time.

113 Heuristic expiration
MUST be included if the cache heuristically chose a freshness lifetime greater than 24 hours and the response's age is greater than 24 hours.

199 Miscellaneous warning
The warning text MAY include arbitrary information to be presented to a human user, or logged. A system receiving this warning MUST NOT take any automated action, besides presenting the warning to the user.





214 Transformation applied
MUST be added by an intermediate cache or proxy if it applies any transformation changing the content-coding (as specified in the Content-Encoding header) or media-type (as specified in the Content-Type header) of the response, or the entity-body of the response, unless this Warning code already appears in the response.

299 Miscellaneous persistent warning
The warning text MAY include arbitrary information to be presented to a human user, or logged. A system receiving this warning MUST NOT take any automated action.

If an implementation sends a message with one or more Warning headers whose version is HTTP/1.0 or lower, then the sender MUST include in each warning-value a warn-date that matches the date in the response.

If an implementation receives a message with a warning-value that includes a warn-date, and that warn-date is different from the Date value in the response, then that warning-value MUST be deleted from the message before storing, forwarding, or using it. (This prevents bad consequences of naive caching of Warning header fields.) If all of the warning-values are deleted for this reason, the Warning header MUST be deleted as well.

### 14.47 WWW-Authenticate

The WWW-Authenticate response-header field MUST be included in 401 (Unauthorized) response messages. The field value consists of at

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

least one challenge that indicates the authentication scheme(s) and parameters applicable to the Request-URI.

```
    WWW-Authenticate  = "WWW-Authenticate" ":" 1#challenge
```

The HTTP access authentication process is described in "HTTP Authentication: Basic and Digest Access Authentication" [43]. User agents are advised to take special care in parsing the WWW-Authenticate field value as it might contain more than one challenge, or if more than one WWW-Authenticate header field is provided, the contents of a challenge itself can contain a comma-separated list of authentication parameters.

## 15 Security Considerations

This section is meant to inform application developers, information providers, and users of the security limitations in HTTP/1.1 as described by this document. The discussion does not include definitive solutions to the problems revealed, though it does make some suggestions for reducing security risks.





### 15.1 Personal Information

HTTP clients are often privy to large amounts of personal information (e.g. the user's name, location, mail address, passwords, encryption keys, etc.), and SHOULD be very careful to prevent unintentional leakage of this information via the HTTP protocol to other sources. We very strongly recommend that a convenient interface be provided for the user to control dissemination of such information, and that designers and implementors be particularly careful in this area. History shows that errors in this area often create serious security and/or privacy problems and generate highly adverse publicity for the implementor's company.

#### 15.1.1 Abuse of Server Log Information

A server is in the position to save personal data about a user's requests which might identify their reading patterns or subjects of interest. This information is clearly confidential in nature and its handling can be constrained by law in certain countries. People using the HTTP protocol to provide data are responsible for ensuring that such material is not distributed without the permission of any individuals that are identifiable by the published results.

#### 15.1.2 Transfer of Sensitive Information

Like any generic data transfer protocol, HTTP cannot regulate the content of the data that is transferred, nor is there any a priori method of determining the sensitivity of any particular piece of information within the context of any given request. Therefore, applications SHOULD supply as much control over this information as possible to the provider of that information. Four header fields are worth special mention in this context: Server, Via, Referer and From.

Revealing the specific software version of the server might allow the server machine to become more vulnerable to attacks against software

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1211
Keysight, EX-1045
Page 1211

that is known to contain security holes. Implementors SHOULD make the Server header field a configurable option.

Proxies which serve as a portal through a network firewall SHOULD take special precautions regarding the transfer of header information that identifies the hosts behind the firewall. In particular, they SHOULD remove, or replace with sanitized versions, any Via fields generated behind the firewall.

The Referer header allows reading patterns to be studied and reverse links drawn. Although it can be very useful, its power can be abused if user details are not separated from the information contained in





the Referer. Even when the personal information has been removed, the Referer header might indicate a private document's URI whose publication would be inappropriate.

The information sent in the From field might conflict with the user's privacy interests or their site's security policy, and hence it SHOULD NOT be transmitted without the user being able to disable, enable, and modify the contents of the field. The user MUST be able to set the contents of this field within a user preference or application defaults configuration.

We suggest, though do not require, that a convenient toggle interface be provided for the user to enable or disable the sending of From and Referer information.

The User-Agent (section 14.43) or Server (section 14.38) header fields can sometimes be used to determine that a specific client or server have a particular security hole which might be exploited. Unfortunately, this same information is often used for other valuable purposes for which HTTP currently has no better mechanism.

### 15.1.3 Encoding Sensitive Information in URI's

Because the source of a link might be private information or might reveal an otherwise private information source, it is strongly recommended that the user be able to select whether or not the Referer field is sent. For example, a browser client could have a toggle switch for browsing openly/anonymously, which would respectively enable/disable the sending of Referer and From information.

Clients SHOULD NOT include a Referer header field in a (non-secure) HTTP request if the referring page was transferred with a secure protocol.

Authors of services which use the HTTP protocol SHOULD NOT use GET based forms for the submission of sensitive data, because this will cause this data to be encoded in the Request-URI. Many existing servers, proxies, and user agents will log the request URI in some place where it might be visible to third parties. Servers can use POST-based form submission instead

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1212
Keysight, EX-1045
Page 1212

15.1.4 Privacy Issues Connected to Accept Headers

Accept request-headers can reveal information about the user to all servers which are accessed. The Accept-Language header in particular can reveal information the user would consider to be of a private nature, because the understanding of particular languages is often





strongly correlated to the membership of a particular ethnic group. User agents which offer the option to configure the contents of an Accept-Language header to be sent in every request are strongly encouraged to let the configuration process include a message which makes the user aware of the loss of privacy involved.

An approach that limits the loss of privacy would be for a user agent to omit the sending of Accept-Language headers by default, and to ask the user whether or not to start sending Accept-Language headers to a server if it detects, by looking for any Vary response-header fields generated by the server, that such sending could improve the quality of service.

Elaborate user-customized accept header fields sent in every request, in particular if these include quality values, can be used by servers as relatively reliable and long-lived user identifiers. Such user identifiers would allow content providers to do click-trail tracking, and would allow collaborating content providers to match cross-server click-trails or form submissions of individual users. Note that for many users not behind a proxy, the network address of the host running the user agent will also serve as a long-lived user identifier. In environments where proxies are used to enhance privacy, user agents ought to be conservative in offering accept header configuration options to end users. As an extreme privacy measure, proxies could filter the accept headers in relayed requests. General purpose user agents which provide a high degree of header configurability SHOULD warn users about the loss of privacy which can be involved.

15.2 Attacks Based On File and Path Names

Implementations of HTTP origin servers SHOULD be careful to restrict the documents returned by HTTP requests to be only those that were intended by the server administrators. If an HTTP server translates HTTP URIs directly into file system calls, the server MUST take special care not to serve files that were not intended to be delivered to HTTP clients. For example, UNIX, Microsoft Windows, and other operating systems use ".." as a path component to indicate a directory level above the current one. On such a system, an HTTP server MUST disallow any such construct in the Request-URI if it would otherwise allow access to a resource outside those intended to be accessible via the HTTP server. Similarly, files intended for reference only internally to the server (such as access control files, configuration files, and script code) MUST be protected from inappropriate retrieval, since they might contain sensitive information. Experience has shown that minor bugs in such HTTP server implementations have turned into security risks.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

### 15.3 DNS Spoofing

Clients using HTTP rely heavily on the Domain Name Service, and are thus generally prone to security attacks based on the deliberate mis-association of IP addresses and DNS names. Clients need to be cautious in assuming the continuing validity of an IP number/DNS name association.

In particular, HTTP clients SHOULD rely on their name resolver for confirmation of an IP number/DNS name association, rather than caching the result of previous host name lookups. Many platforms already can cache host name lookups locally when appropriate, and they SHOULD be configured to do so. It is proper for these lookups to be cached, however, only when the TTL (Time To Live) information reported by the name server makes it likely that the cached information will remain useful.

If HTTP clients cache the results of host name lookups in order to achieve a performance improvement, they MUST observe the TTL information reported by DNS.

If HTTP clients do not observe this rule, they could be spoofed when a previously-accessed server's IP address changes. As network renumbering is expected to become increasingly common [24], the possibility of this form of attack will grow. Observing this requirement thus reduces this potential security vulnerability.

This requirement also improves the load-balancing behavior of clients for replicated servers using the same DNS name and reduces the likelihood of a user's experiencing failure in accessing sites which use that strategy.

### 15.4 Location Headers and Spoofing

If a single server supports multiple organizations that do not trust one another, then it MUST check the values of Location and Content-Location headers in responses that are generated under control of said organizations to make sure that they do not attempt to invalidate resources over which they have no authority.

### 15.5 Content-Disposition Issues

RFC 1806 [35], from which the often implemented Content-Disposition (see section 19.5.1) header in HTTP is derived, has a number of very serious security considerations. Content-Disposition is not part of the HTTP standard, but since it is widely implemented, we are documenting its use and risks for implementors. See RFC 2183 [49] (which updates RFC 1806) for details.



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

### 15.6 Authentication Credentials and Idle Clients

Existing HTTP clients and user agents typically retain authentication information indefinitely. HTTP/1.1. does not provide a method for a server to direct clients to discard these cached credentials. This is a significant defect that requires further extensions to HTTP. Circumstances under which credential caching can interfere with the application's security model include but are not limited to:

- Clients which have been idle for an extended period following which the server might wish to cause the client to reprompt the user for credentials.

- Applications which include a session termination indication (such as a `logout' or `commit' button on a page) after which the server side of the application `knows' that there is no further reason for the client to retain the credentials.

This is currently under separate study. There are a number of work-arounds to parts of this problem, and we encourage the use of password protection in screen savers, idle time-outs, and other methods which mitigate the security problems inherent in this problem. In particular, user agents which cache credentials are encouraged to provide a readily accessible mechanism for discarding cached credentials under user control.

### 15.7 Proxies and Caching

By their very nature, HTTP proxies are men-in-the-middle, and represent an opportunity for man-in-the-middle attacks. Compromise of the systems on which the proxies run can result in serious security and privacy problems. Proxies have access to security-related information, personal information about individual users and organizations, and proprietary information belonging to users and content providers. A compromised proxy, or a proxy implemented or configured without regard to security and privacy considerations, might be used in the commission of a wide range of potential attacks.

Proxy operators should protect the systems on which proxies run as they would protect any system that contains or transports sensitive information. In particular, log information gathered at proxies often contains highly sensitive personal information, and/or information about organizations. Log information should be carefully guarded, and appropriate guidelines for use developed and followed. (Section 15.1.1).





Caching proxies provide additional potential vulnerabilities, since the contents of the cache represent an attractive target for malicious exploitation. Because cache contents persist after an HTTP request is complete, an attack on the cache can reveal information long after a user believes that the information has been removed from the network. Therefore, cache contents should be protected as sensitive information.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1215
Keysight, EX-1045
Page 1215

Proxy implementors should consider the privacy and security implications of their design and coding decisions, and of the configuration options they provide to proxy operators (especially the default configuration).

Users of a proxy need to be aware that they are no trustworthier than the people who run the proxy; HTTP itself cannot solve this problem.

The judicious use of cryptography, when appropriate, may suffice to protect against a broad range of security and privacy attacks. Such cryptography is beyond the scope of the HTTP/1.1 specification.

### 15.7.1 Denial of Service Attacks on Proxies

They exist. They are hard to defend against. Research continues. Beware.

## 16 Acknowledgments

This specification makes heavy use of the augmented BNF and generic constructs defined by David H. Crocker for RFC 822 [9]. Similarly, it reuses many of the definitions provided by Nathaniel Borenstein and Ned Freed for MIME [7]. We hope that their inclusion in this specification will help reduce past confusion over the relationship between HTTP and Internet mail message formats.

The HTTP protocol has evolved considerably over the years. It has benefited from a large and active developer community--the many people who have participated on the www-talk mailing list--and it is that community which has been most responsible for the success of HTTP and of the World-Wide Web in general. Marc Andreessen, Robert Cailliau, Daniel W. Connolly, Bob Denny, John Franks, Jean-Francois Groff, Phillip M. Hallam-Baker, Hakon W. Lie, Ari Luotonen, Rob McCool, Lou Montulli, Dave Raggett, Tony Sanders, and Marc VanHeyningen deserve special recognition for their efforts in defining early aspects of the protocol.

This document has benefited greatly from the comments of all those participating in the HTTP-WG. In addition to those already mentioned, the following individuals have contributed to this specification:





Gary Adams
Harald Tveit Alvestrand
Keith Ball
Brian Behlendorf
Paul Burchard
Maurizio Codogno
Mike Cowlishaw
Roman Czyborra
Michael A. Dolan
David J. Fiander
Alan Freier
Marc Hedlund
Greg Herlihy
Koen Holtman
Ross Patterson
Albert Lunde
John C. Mallery
Jean-Philippe Martin-Flatin
Mitra
David Morris
Gavin Nicol
Bill Perry
Jeffrey Perry
Scott Powers
Owen Rees
Luigi Rizzo
David Robinson
Marc Salomon

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1216
Keysight, EX-1045
Page 1216

| | |
|---|---|
| Alex Hopmann | Rich Salz |
| Bob Jernigan | Allan M. Schiffman |
| Shel Kaphan | Jim Seidman |
| Rohit Khare | Chuck Shotton |
| John Klensin | Eric W. Sink |
| Martijn Koster | Simon E. Spero |
| Alexei Kosut | Richard N. Taylor |
| David M. Kristol | Robert S. Thau |
| Daniel LaLiberte | Bill (BearHeart) Weinman |
| Ben Laurie | Francois Yergeau |
| Paul J. Leach | Mary Ellen Zurko |
| Daniel DuBois | Josh Cohen |

Much of the content and presentation of the caching design is due to suggestions and comments from individuals including: Shel Kaphan, Paul Leach, Koen Holtman, David Morris, and Larry Masinter.

Most of the specification of ranges is based on work originally done by Ari Luotonen and John Franks, with additional input from Steve Zilles.

Thanks to the "cave men" of Palo Alto. You know who you are.

Jim Gettys (the current editor of this document) wishes particularly to thank Roy Fielding, the previous editor of this document, along with John Klensin, Jeff Mogul, Paul Leach, Dave Kristol, Koen Holtman, John Franks, Josh Cohen, Alex Hopmann, Scott Lawrence, and Larry Masinter for their help. And thanks go particularly to Jeff Mogul and Scott Lawrence for performing the "MUST/MAY/SHOULD" audit.





The Apache Group, Anselm Baird-Smith, author of Jigsaw, and Henrik Frystyk implemented RFC 2068 early, and we wish to thank them for the discovery of many of the problems that this document attempts to rectify.

## 17 References

[1] Alvestrand, H., "Tags for the Identification of Languages", RFC 1766, March 1995.

[2] Anklesaria, F., McCahill, M., Lindner, P., Johnson, D., Torrey, D. and B. Alberti, "The Internet Gopher Protocol (a distributed document search and retrieval protocol)", RFC 1436, March 1993.

[3] Berners-Lee, T., "Universal Resource Identifiers in WWW", RFC 1630, June 1994.

[4] Berners-Lee, T., Masinter, L. and M. McCahill, "Uniform Resource Locators (URL)", RFC 1738, December 1994.

[5] Berners-Lee, T. and D. Connolly, "Hypertext Markup Language -

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1217
Keysight, EX-1045
Page 1217

2.0", RFC 1866, November 1995.

[6] Berners-Lee, T., Fielding, R. and H. Frystyk, "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996.

[7] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[8] Braden, R., "Requirements for Internet Hosts -- Communication Layers", STD 3, RFC 1123, October 1989.

[9] Crocker, D., "Standard for The Format of ARPA Internet Text Messages", STD 11, RFC 822, August 1982.

[10] Davis, F., Kahle, B., Morris, H., Salem, J., Shen, T., Wang, R., Sui, J., and M. Grinbaum, "WAIS Interface Protocol Prototype Functional Specification," (v1.5), Thinking Machines Corporation, April 1990.

[11] Fielding, R., "Relative Uniform Resource Locators", RFC 1808, June 1995.

[12] Horton, M. and R. Adams, "Standard for Interchange of USENET Messages", RFC 1036, December 1987.





[13] Kantor, B. and P. Lapsley, "Network News Transfer Protocol", RFC 977, February 1986.

[14] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.

[15] Nebel, E. and L. Masinter, "Form-based File Upload in HTML", RFC 1867, November 1995.

[16] Postel, J., "Simple Mail Transfer Protocol", STD 10, RFC 821, August 1982.

[17] Postel, J., "Media Type Registration Procedure", RFC 1590, November 1996.

[18] Postel, J. and J. Reynolds, "File Transfer Protocol", STD 9, RFC 959, October 1985.

[19] Reynolds, J. and J. Postel, "Assigned Numbers", STD 2, RFC 1700, October 1994.

[20] Sollins, K. and L. Masinter, "Functional Requirements for Uniform Resource Names", RFC 1737, December 1994.

[21] US-ASCII. Coded Character Set - 7-Bit American Standard Code for Information Interchange. Standard ANSI X3.4-1986, ANSI, 1986.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1218
Keysight, EX-1045
Page 1218

[22] ISO-8859. International Standard -- Information Processing --
8-bit Single-Byte Coded Graphic Character Sets --
Part 1: Latin alphabet No. 1, ISO-8859-1:1987.
Part 2: Latin alphabet No. 2, ISO-8859-2, 1987.
Part 3: Latin alphabet No. 3, ISO-8859-3, 1988.
Part 4: Latin alphabet No. 4, ISO-8859-4, 1988.
Part 5: Latin/Cyrillic alphabet, ISO-8859-5, 1988.
Part 6: Latin/Arabic alphabet, ISO-8859-6, 1987.
Part 7: Latin/Greek alphabet, ISO-8859-7, 1987.
Part 8: Latin/Hebrew alphabet, ISO-8859-8, 1988.
Part 9: Latin alphabet No. 5, ISO-8859-9, 1990.

[23] Meyers, J. and M. Rose, "The Content-MD5 Header Field", RFC 1864, October 1995.

[24] Carpenter, B. and Y. Rekhter, "Renumbering Needs Work", RFC 1900, February 1996.

[25] Deutsch, P., "GZIP file format specification version 4.3", RFC 1952, May 1996.





[26] Venkata N. Padmanabhan, and Jeffrey C. Mogul. "Improving HTTP Latency", Computer Networks and ISDN Systems, v. 28, pp. 25-35, Dec. 1995. Slightly revised version of paper in Proc. 2nd International WWW Conference '94: Mosaic and the Web, Oct. 1994, which is available at http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/DDay/mogul/HTTPLatency.html.

[27] Joe Touch, John Heidemann, and Katia Obraczka. "Analysis of HTTP Performance", <URL: http://www.isi.edu/touch/pubs/http-perf96/>, ISI Research Report ISI/RR-98-463, (original report dated Aug. 1996), USC/Information Sciences Institute, August 1998.

[28] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation and Analysis", RFC 1305, March 1992.

[29] Deutsch, P., "DEFLATE Compressed Data Format Specification version 1.3", RFC 1951, May 1996.

[30] S. Spero, "Analysis of HTTP Performance Problems," http://sunsite.unc.edu/mdma-release/http-prob.html.

[31] Deutsch, P. and J. Gailly, "ZLIB Compressed Data Format Specification version 3.3", RFC 1950, May 1996.

[32] Franks, J., Hallam-Baker, P., Hostetler, J., Leach, P., Luotonen, A., Sink, E. and L. Stewart, "An Extension to HTTP: Digest Access Authentication", RFC 2069, January 1997.

[33] Fielding, R., Gettys, J., Mogul, J., Frystyk, H. and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2068, January 1997.

[34] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1219
Keysight, EX-1045
Page 1219

[35] Troost, R. and Dorner, S., "Communicating Presentation Information in Internet Messages: The Content-Disposition Header", RFC 1806, June 1995.

[36] Mogul, J., Fielding, R., Gettys, J. and H. Frystyk, "Use and Interpretation of HTTP Version Numbers", RFC 2145, May 1997. [jg639]

[37] Palme, J., "Common Internet Message Headers", RFC 2076, February 1997. [jg640]





[38] Yergeau, F., "UTF-8, a transformation format of Unicode and ISO-10646", RFC 2279, January 1998. [jg641]

[39] Nielsen, H.F., Gettys, J., Baird-Smith, A., Prud'hommeaux, E., Lie, H., and C. Lilley. "Network Performance Effects of HTTP/1.1, CSS1, and PNG," Proceedings of ACM SIGCOMM '97, Cannes France, September 1997.[jg642]

[40] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996. [jg643]

[41] Alvestrand, H., "IETF Policy on Character Sets and Languages", BCP 18, RFC 2277, January 1998. [jg644]

[42] Berners-Lee, T., Fielding, R. and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax and Semantics", RFC 2396, August 1998. [jg645]

[43] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., Sink, E. and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999. [jg646]

[44] Luotonen, A., "Tunneling TCP based protocols through Web proxy servers," Work in Progress. [jg647]

[45] Palme, J. and A. Hopmann, "MIME E-mail Encapsulation of Aggregate Documents, such as HTML (MHTML)", RFC 2110, March 1997.

[46] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[47] Masinter, L., "Hyper Text Coffee Pot Control Protocol (HTCPCP/1.0)", RFC 2324, 1 April 1998.

[48] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples", RFC 2049, November 1996.

[49] Troost, R., Dorner, S. and K. Moore, "Communicating Presentation

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1220
Keysight, EX-1045
Page 1220

Information in Internet Messages: The Content-Disposition Header
Field", RFC 2183, August 1997.





## 18 Authors' Addresses

Roy T. Fielding
Information and Computer Science
University of California, Irvine
Irvine, CA 92697-3425, USA

Fax: +1 (949) 824-1715
EMail: fielding@ics.uci.edu

James Gettys
World Wide Web Consortium
MIT Laboratory for Computer Science
545 Technology Square
Cambridge, MA 02139, USA

Fax: +1 (617) 258 8682
EMail: jg@w3.org

Jeffrey C. Mogul
Western Research Laboratory
Compaq Computer Corporation
250 University Avenue
Palo Alto, California, 94305, USA

EMail: mogul@wrl.dec.com

Henrik Frystyk Nielsen
World Wide Web Consortium
MIT Laboratory for Computer Science
545 Technology Square
Cambridge, MA 02139, USA

Fax: +1 (617) 258 8682
EMail: frystyk@w3.org

Larry Masinter
Xerox Corporation
3333 Coyote Hill Road
Palo Alto, CA 94034, USA

EMail: masinter@parc.xerox.com

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1221
Keysight, EX-1045
Page 1221

Paul J. Leach
Microsoft Corporation
1 Microsoft Way
Redmond, WA 98052, USA

EMail: paulle@microsoft.com

Tim Berners-Lee
Director, World Wide Web Consortium
MIT Laboratory for Computer Science
545 Technology Square
Cambridge, MA 02139, USA

Fax: +1 (617) 258 8682
EMail: timbl@w3.org



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1222
Keysight, EX-1045
Page 1222

# 19 Appendices

## 19.1 Internet Media Type message/http and application/http

In addition to defining the HTTP/1.1 protocol, this document serves as the specification for the Internet media type "message/http" and "application/http". The message/http type can be used to enclose a single HTTP request or response message, provided that it obeys the MIME restrictions for all "message" types regarding line length and encodings. The application/http type can be used to enclose a pipeline of one or more HTTP request or response messages (not intermixed). The following is to be registered with IANA [17].

```
    Media Type name:         message
    Media subtype name:      http
    Required parameters:     none
    Optional parameters:     version, msgtype
     version: The HTTP-Version number of the enclosed message
              (e.g., "1.1"). If not present, the version can be
              determined from the first line of the body.
     msgtype: The message type -- "request" or "response". If not
              present, the type can be determined from the first
              line of the body.
    Encoding considerations: only "7bit", "8bit", or "binary" are
                             permitted
    Security considerations: none

    Media Type name:         application
    Media subtype name:      http
    Required parameters:     none
    Optional parameters:     version, msgtype
     version: The HTTP-Version number of the enclosed messages
              (e.g., "1.1"). If not present, the version can be
              determined from the first line of the body.
     msgtype: The message type -- "request" or "response". If not
              present, the type can be determined from the first
              line of the body.
    Encoding considerations: HTTP messages enclosed by this type
              are in "binary" format; use of an appropriate
              Content-Transfer-Encoding is required when
              transmitted via E-mail.
    Security considerations: none
```





## 19.2 Internet Media Type multipart/byteranges

When an HTTP 206 (Partial Content) response message includes the content of multiple ranges (a response to a request for multiple non-overlapping ranges), these are transmitted as a multipart message-body. The media type for this purpose is called "multipart/byteranges".

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The multipart/byteranges media type includes two or more parts, each with its own Content-Type and Content-Range fields. The required boundary parameter specifies the boundary string used to separate each body-part.

```
    Media Type name:         multipart
    Media subtype name:      byteranges
    Required parameters:     boundary
    Optional parameters:     none
    Encoding considerations: only "7bit", "8bit", or "binary" are
                             permitted
    Security considerations: none
```

For example:

```
HTTP/1.1 206 Partial Content
Date: Wed, 15 Nov 1995 06:25:24 GMT
Last-Modified: Wed, 15 Nov 1995 04:58:08 GMT
Content-type: multipart/byteranges; boundary=THIS_STRING_SEPARATES

--THIS_STRING_SEPARATES
Content-type: application/pdf
Content-range: bytes 500-999/8000

...the first range...
--THIS_STRING_SEPARATES
Content-type: application/pdf
Content-range: bytes 7000-7999/8000

...the second range
--THIS_STRING_SEPARATES--
```

Notes:

1) Additional CRLFs may precede the first boundary string in the entity.





2) Although RFC 2046 [40] permits the boundary string to be quoted, some existing implementations handle a quoted boundary string incorrectly.

3) A number of browsers and servers were coded to an early draft of the byteranges specification to use a media type of multipart/x-byteranges, which is almost, but not quite compatible with the version documented in HTTP/1.1.

## 19.3 Tolerant Applications

Although this document specifies the requirements for the generation of HTTP/1.1 messages, not all applications will be correct in their implementation. We therefore recommend that operational applications

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

be tolerant of deviations whenever those deviations can be interpreted unambiguously.

Clients SHOULD be tolerant in parsing the Status-Line and servers tolerant when parsing the Request-Line. In particular, they SHOULD accept any amount of SP or HT characters between fields, even though only a single SP is required.

The line terminator for message-header fields is the sequence CRLF. However, we recommend that applications, when parsing such headers, recognize a single LF as a line terminator and ignore the leading CR.

The character set of an entity-body SHOULD be labeled as the lowest common denominator of the character codes used within that body, with the exception that not labeling the entity is preferred over labeling the entity with the labels US-ASCII or ISO-8859-1. See section 3.7.1 and 3.4.1.

Additional rules for requirements on parsing and encoding of dates and other potential problems with date encodings include:

- HTTP/1.1 clients and caches SHOULD assume that an RFC-850 date which appears to be more than 50 years in the future is in fact in the past (this helps solve the "year 2000" problem).

- An HTTP/1.1 implementation MAY internally represent a parsed Expires date as earlier than the proper value, but MUST NOT internally represent a parsed Expires date as later than the proper value.

- All expiration-related calculations MUST be done in GMT. The local time zone MUST NOT influence the calculation or comparison of an age or expiration time.





- If an HTTP header incorrectly carries a date value with a time zone other than GMT, it MUST be converted into GMT using the most conservative possible conversion.

## 19.4 Differences Between HTTP Entities and RFC 2045 Entities

HTTP/1.1 uses many of the constructs defined for Internet Mail (RFC 822 [9]) and the Multipurpose Internet Mail Extensions (MIME [7]) to allow entities to be transmitted in an open variety of representations and with extensible mechanisms. However, RFC 2045 discusses mail, and HTTP has a few features that are different from those described in RFC 2045. These differences were carefully chosen to optimize performance over binary connections, to allow greater freedom in the use of new media types, to make date comparisons easier, and to acknowledge the practice of some early HTTP servers and clients.

This appendix describes specific areas where HTTP differs from RFC 2045. Proxies and gateways to strict MIME environments SHOULD be aware of these differences and provide the appropriate conversions where necessary. Proxies and gateways from MIME environments to HTTP

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

also need to be aware of the differences because some conversions might be required.

### 19.4.1 MIME-Version

HTTP is not a MIME-compliant protocol. However, HTTP/1.1 messages MAY include a single MIME-Version general-header field to indicate what version of the MIME protocol was used to construct the message. Use of the MIME-Version header field indicates that the message is in full compliance with the MIME protocol (as defined in RFC 2045[7]). Proxies/gateways are responsible for ensuring full compliance (where possible) when exporting HTTP messages to strict MIME environments.

```
    MIME-Version   = "MIME-Version" ":" 1*DIGIT "." 1*DIGIT
```

MIME version "1.0" is the default for use in HTTP/1.1. However, HTTP/1.1 message parsing and semantics are defined by this document and not the MIME specification.

### 19.4.2 Conversion to Canonical Form

RFC 2045 [7] requires that an Internet mail entity be converted to canonical form prior to being transferred, as described in section 4 of RFC 2049 [48]. Section 3.7.1 of this document describes the forms allowed for subtypes of the "text" media type when transmitted over HTTP. RFC 2046 requires that content with a type of "text" represent line breaks as CRLF and forbids the use of CR or LF outside of line





break sequences. HTTP allows CRLF, bare CR, and bare LF to indicate a line break within text content when a message is transmitted over HTTP.

Where it is possible, a proxy or gateway from HTTP to a strict MIME environment SHOULD translate all line breaks within the text media types described in section 3.7.1 of this document to the RFC 2049 canonical form of CRLF. Note, however, that this might be complicated by the presence of a Content-Encoding and by the fact that HTTP allows the use of some character sets which do not use octets 13 and 10 to represent CR and LF, as is the case for some multi-byte character sets.

Implementors should note that conversion will break any cryptographic checksums applied to the original content unless the original content is already in canonical form. Therefore, the canonical form is recommended for any content that uses such checksums in HTTP.

### 19.4.3 Conversion of Date Formats

HTTP/1.1 uses a restricted set of date formats (section 3.3.1) to simplify the process of date comparison. Proxies and gateways from other protocols SHOULD ensure that any Date header field present in a message conforms to one of the HTTP/1.1 formats and rewrite the date if necessary.

### 19.4.4 Introduction of Content-Encoding

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

RFC 2045 does not include any concept equivalent to HTTP/1.1's Content-Encoding header field. Since this acts as a modifier on the media type, proxies and gateways from HTTP to MIME-compliant protocols MUST either change the value of the Content-Type header field or decode the entity-body before forwarding the message. (Some experimental applications of Content-Type for Internet mail have used a media-type parameter of ";conversions=<content-coding>" to perform a function equivalent to Content-Encoding. However, this parameter is not part of RFC 2045.)

### 19.4.5 No Content-Transfer-Encoding

HTTP does not use the Content-Transfer-Encoding (CTE) field of RFC 2045. Proxies and gateways from MIME-compliant protocols to HTTP MUST remove any non-identity CTE ("quoted-printable" or "base64") encoding prior to delivering the response message to an HTTP client.

Proxies and gateways from HTTP to MIME-compliant protocols are responsible for ensuring that the message is in the correct format and encoding for safe transport on that protocol, where "safe





transport" is defined by the limitations of the protocol being used. Such a proxy or gateway SHOULD label the data with an appropriate Content-Transfer-Encoding if doing so will improve the likelihood of safe transport over the destination protocol.

### 19.4.6 Introduction of Transfer-Encoding

HTTP/1.1 introduces the Transfer-Encoding header field (section 14.41). Proxies/gateways MUST remove any transfer-coding prior to forwarding a message via a MIME-compliant protocol.

A process for decoding the "chunked" transfer-coding (section 3.6) can be represented in pseudo-code as:

```
length := 0
read chunk-size, chunk-extension (if any) and CRLF
while (chunk-size > 0) {
   read chunk-data and CRLF
   append chunk-data to entity-body
   length := length + chunk-size
   read chunk-size and CRLF
}
read entity-header
while (entity-header not empty) {
   append entity-header to existing header fields
   read entity-header
}
Content-Length := length
Remove "chunked" from Transfer-Encoding
```

### 19.4.7 MHTML and Line Length Limitations

HTTP implementations which share code with MHTML [45] implementations need to be aware of MIME line length limitations. Since HTTP does not have this limitation, HTTP does not fold long lines. MHTML messages

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1227
Keysight, EX-1045
Page 1227

being transported by HTTP follow all conventions of MHTML, including line length limitations and folding, canonicalization, etc., since HTTP transports all message-bodies as payload (see section 3.7.2) and does not interpret the content or any MIME header lines that might be contained therein.

### 19.5 Additional Features

RFC 1945 and RFC 2068 document protocol elements used by some existing HTTP implementations, but not consistently and correctly across most HTTP/1.1 applications. Implementors are advised to be aware of these features, but cannot rely upon their presence in, or interoperability with, other HTTP/1.1 applications. Some of these





describe proposed experimental features, and some describe features that experimental deployment found lacking that are now addressed in the base HTTP/1.1 specification.

A number of other headers, such as Content-Disposition and Title, from SMTP and MIME are also often implemented (see RFC 2076 [37]).

#### 19.5.1 Content-Disposition

The Content-Disposition response-header field has been proposed as a means for the origin server to suggest a default filename if the user requests that the content is saved to a file. This usage is derived from the definition of Content-Disposition in RFC 1806 [35].

```
content-disposition = "Content-Disposition" ":"
                      disposition-type *( ";" disposition-parm )
disposition-type = "attachment" | disp-extension-token
disposition-parm = filename-parm | disp-extension-parm
filename-parm = "filename" "=" quoted-string
disp-extension-token = token
disp-extension-parm = token "=" ( token | quoted-string )
```

An example is

```
Content-Disposition: attachment; filename="fname.ext"
```

The receiving user agent SHOULD NOT respect any directory path information present in the filename-parm parameter, which is the only parameter believed to apply to HTTP implementations at this time. The filename SHOULD be treated as a terminal component only.

If this header is used in a response with the application/octet-stream content-type, the implied suggestion is that the user agent should not display the response, but directly enter a `save response as...' dialog.

See section 15.5 for Content-Disposition security issues.

### 19.6 Compatibility with Previous Versions

It is beyond the scope of a protocol specification to mandate compliance with previous versions. HTTP/1.1 was deliberately

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

designed, however, to make supporting previous versions easy. It is worth noting that, at the time of composing this specification (1996), we would expect commercial HTTP/1.1 servers to:

- recognize the format of the Request-Line for HTTP/0.9, 1.0, and 1.1 requests;





- understand any valid request in the format of HTTP/0.9, 1.0, or 1.1;

- respond appropriately with a message in the same major version used by the client.

And we would expect HTTP/1.1 clients to:

- recognize the format of the Status-Line for HTTP/1.0 and 1.1 responses;

- understand any valid response in the format of HTTP/0.9, 1.0, or 1.1.

For most implementations of HTTP/1.0, each connection is established by the client prior to the request and closed by the server after sending the response. Some implementations implement the Keep-Alive version of persistent connections described in section 19.7.1 of RFC 2068 [33].

### 19.6.1 Changes from HTTP/1.0

This section summarizes major differences between versions HTTP/1.0 and HTTP/1.1.

#### 19.6.1.1 Changes to Simplify Multi-homed Web Servers and Conserve IP Addresses

The requirements that clients and servers support the Host request-header, report an error if the Host request-header (section 14.23) is missing from an HTTP/1.1 request, and accept absolute URIs (section 5.1.2) are among the most important changes defined by this specification.

Older HTTP/1.0 clients assumed a one-to-one relationship of IP addresses and servers; there was no other established mechanism for distinguishing the intended server of a request than the IP address to which that request was directed. The changes outlined above will allow the Internet, once older HTTP clients are no longer common, to support multiple Web sites from a single IP address, greatly simplifying large operational Web servers, where allocation of many IP addresses to a single host has created serious problems. The Internet will also be able to recover the IP addresses that have been allocated for the sole purpose of allowing special-purpose domain names to be used in root-level HTTP URLs. Given the rate of growth of the Web, and the number of servers already deployed, it is extremely

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

important that all implementations of HTTP (including updates to existing HTTP/1.0 applications) correctly implement these requirements:

- Both clients and servers MUST support the Host request-header.

- A client that sends an HTTP/1.1 request MUST send a Host header.

- Servers MUST report a 400 (Bad Request) error if an HTTP/1.1 request does not include a Host request-header.

- Servers MUST accept absolute URIs.

### 19.6.2 Compatibility with HTTP/1.0 Persistent Connections

Some clients and servers might wish to be compatible with some previous implementations of persistent connections in HTTP/1.0 clients and servers. Persistent connections in HTTP/1.0 are explicitly negotiated as they are not the default behavior. HTTP/1.0 experimental implementations of persistent connections are faulty, and the new facilities in HTTP/1.1 are designed to rectify these problems. The problem was that some existing 1.0 clients may be sending Keep-Alive to a proxy server that doesn't understand Connection, which would then erroneously forward it to the next inbound server, which would establish the Keep-Alive connection and result in a hung HTTP/1.0 proxy waiting for the close on the response. The result is that HTTP/1.0 clients must be prevented from using Keep-Alive when talking to proxies.

However, talking to proxies is the most important use of persistent connections, so that prohibition is clearly unacceptable. Therefore, we need some other mechanism for indicating a persistent connection is desired, which is safe to use even when talking to an old proxy that ignores Connection. Persistent connections are the default for HTTP/1.1 messages; we introduce a new keyword (Connection: close) for declaring non-persistence. See section 14.10.

The original HTTP/1.0 form of persistent connections (the Connection: Keep-Alive and Keep-Alive header) is documented in RFC 2068. [33]

### 19.6.3 Changes from RFC 2068

This specification has been carefully audited to correct and disambiguate key word usage; RFC 2068 had many problems in respect to the conventions laid out in RFC 2119 [34].

Clarified which error code should be used for inbound server failures (e.g. DNS failures). (Section 10.5.5).



https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CREATE had a race that required an Etag be sent when a resource is first created. (Section 10.2.2).

Content-Base was deleted from the specification: it was not implemented widely, and there is no simple, safe way to introduce it without a robust extension mechanism. In addition, it is used in a similar, but not identical fashion in MHTML [45].

Transfer-coding and message lengths all interact in ways that required fixing exactly when chunked encoding is used (to allow for transfer encoding that may not be self delimiting); it was important to straighten out exactly how message lengths are computed. (Sections 3.6, 4.4, 7.2.2, 13.5.2, 14.13, 14.16)

A content-coding of "identity" was introduced, to solve problems discovered in caching. (section 3.5)

Quality Values of zero should indicate that "I don't want something" to allow clients to refuse a representation. (Section 3.9)

The use and interpretation of HTTP version numbers has been clarified by RFC 2145. Require proxies to upgrade requests to highest protocol version they support to deal with problems discovered in HTTP/1.0 implementations (Section 3.1)

Charset wildcarding is introduced to avoid explosion of character set names in accept headers. (Section 14.2)

A case was missed in the Cache-Control model of HTTP/1.1; s-maxage was introduced to add this missing case. (Sections 13.4, 14.8, 14.9, 14.9.3)

The Cache-Control: max-age directive was not properly defined for responses. (Section 14.9.3)

There are situations where a server (especially a proxy) does not know the full length of a response but is capable of serving a byterange request. We therefore need a mechanism to allow byteranges with a content-range not indicating the full length of the message. (Section 14.16)

Range request responses would become very verbose if all meta-data were always returned; by allowing the server to only send needed headers in a 206 response, this problem can be avoided. (Section 10.2.7, 13.5.3, and 14.27)





Fix problem with unsatisfiable range requests; there are two cases: syntactic problems, and range doesn't exist in the document. The 416 status code was needed to resolve this ambiguity needed to indicate an error for a byte range request that falls outside of the actual contents of a document. (Section 10.4.17, 14.16)

Rewrite of message transmission requirements to make it much harder

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1231
Keysight, EX-1045
Page 1231

for implementors to get it wrong, as the consequences of errors here can have significant impact on the Internet, and to deal with the following problems:

1. Changing "HTTP/1.1 or later" to "HTTP/1.1", in contexts where this was incorrectly placing a requirement on the behavior of an implementation of a future version of HTTP/1.x

2. Made it clear that user-agents should retry requests, not "clients" in general.

3. Converted requirements for clients to ignore unexpected 100 (Continue) responses, and for proxies to forward 100 responses, into a general requirement for 1xx responses.

4. Modified some TCP-specific language, to make it clearer that non-TCP transports are possible for HTTP.

5. Require that the origin server MUST NOT wait for the request body before it sends a required 100 (Continue) response.

6. Allow, rather than require, a server to omit 100 (Continue) if it has already seen some of the request body.

7. Allow servers to defend against denial-of-service attacks and broken clients.

This change adds the Expect header and 417 status code. The message transmission requirements fixes are in sections 8.2, 10.4.18, 8.1.2.2, 13.11, and 14.20.

Proxies should be able to add Content-Length when appropriate. (Section 13.5.2)

Clean up confusion between 403 and 404 responses. (Section 10.4.4, 10.4.5, and 10.4.11)

Warnings could be cached incorrectly, or not updated appropriately. (Section 13.1.2, 13.2.4, 13.5.2, 13.5.3, 14.9.3, and 14.46) Warning also needed to be a general header, as PUT or other methods may have need for it in requests.





Transfer-coding had significant problems, particularly with interactions with chunked encoding. The solution is that transfer-codings become as full fledged as content-codings. This involves adding an IANA registry for transfer-codings (separate from content codings), a new header field (TE) and enabling trailer headers in the future. Transfer encoding is a major performance benefit, so it was worth fixing [39]. TE also solves another, obscure, downward interoperability problem that could have occurred due to interactions between authentication trailers, chunked encoding and HTTP/1.0 clients.(Section 3.6, 3.6.1, and 14.39)

The PATCH, LINK, UNLINK methods were defined but not commonly implemented in previous versions of this specification. See RFC 2068 [33].

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

The Alternates, Content-Version, Derived-From, Link, URI, Public and Content-Base header fields were defined in previous versions of this specification, but not commonly implemented. See RFC 2068 [33].

## 20 Index

Please see the PostScript version of this RFC for the INDEX.





## 21. Full Copyright Statement

Copyright (C) The Internet Society (1999). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1233
**Keysight, EX-1045**
**Page 1233**

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.





https://web.archive.org/web/20000815082940if_/http://www.ietf.org:80/rfc/rfc2616.txt

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1234
Keysight, EX-1045
Page 1234

https://web.archive.org/web/20081207003303/http://www.sonicwall.com/downloads/EC_GMS_DS_US.pdf

# SonicWALL® Global Management System

POLICY AND MANAGEMENT

## Centralized Network Monitoring and Management Solution

- **Centralized security and network management**
- **Sophisticated VPN deployment and configuration**
- **Active device monitoring and alerting**
- **Centralized logging**
- **Intelligent reporting and activity visualization**
- **Offline management**
- **Streamlined license manager**
- **SNMP support**

The SonicWALL® Global Management System (GMS) provides organizations, distributed enterprises and service providers with a flexible, powerful and intuitive solution to centrally manage and rapidly deploy SonicWALL appliances and security policy configurations. SonicWALL GMS™ also provides centralized real-time monitoring, and comprehensive policy and compliance reporting.

SonicWALL GMS' intuitive Web-based user interface easily allows for complete life cycle control of thousands of SonicWALL appliances—from initial configuration to complex policy changes and remote updates. For enterprises, GMS simplifies the complexity of network management by offering a single management interface, thereby reducing administration time, complexity and the overall total cost of ownership (TCO). Service providers benefit from its multi-organizational management capabilities where it consolidates, groups and classifies thousands of individual customers' managed appliances and their respective security policies. Through an integrated reporting architecture, administrators can customize and schedule reports individually tailored to meet the needs of managed customers, executives and regulatory compliance audits for corporate departments.

### Features and Benefits

**Centralized security and network management** is achieved using a flexible, powerful and intuitive tool to deploy, manage and monitor a distributed network environment and set policies from a central location. Administrators can now define, distribute, enforce and deploy a full range of service and security policies for thousands of SonicWALL appliances.

**Sophisticated VPN deployment and configuration** enables distributed enterprise networks to reduce the administration time, costs and complexity associated with establishing and maintaining corporate security policies, VPN connectivity and network configurations. For Service Providers, it consolidates and unifies all security polices for thousands of customers allowing for greater efficiencies in delivering an SLA.

**Active device monitoring and alerting** supplies real-time alerts with integrated monitoring capabilities allowing administrators to take preventative action and deliver immediate remediation.

**Centralized logging** provides a central location for consolidating security events and logs for thousands of appliances, thereby allowing for a single point to conduct network forensics.

**Intelligent reporting and activity visualization** presents comprehensive management and graphical reports for security devices and user activity yielding greater insight into usage trends and security events. It also enables the customization of these reports using corporate logos and colors, thereby delivering a cohesive branding message to users and customers.

**Offline management** enables scheduled configurations and/or firmware updates on managed appliances to occur offline to minimize service disruption to end users and customers.

**Streamlined license manager** displays a unified console for storing, applying, tracking and updating security license information for managed SonicWALL appliances, thereby simplifying the inventory for managing security and support license subscriptions.

**SNMP support** provides a powerful, real-time alerting mechanism for all TCP/IP and SNMP-enabled devices and applications, thus greatly enhancing the ability to pinpoint and respond to critical network events.

SONICWALL®

PROTECTION AT THE SPEED OF BUSINESS®

## Specifications

**SonicWALL GMS Standard Edition**

SonicWALL GMS 10 Node Software License
01-SSC-3363

SonicWALL GMS 25 Node Software License
01-SSC-3311

SonicWALL GMS 24x7 Software Support for 10 Nodes (1-year)*
01-SSC-6514

SonicWALL GMS 24x7 Software Support for 25 Nodes (1-year)*
01-SSC-3334

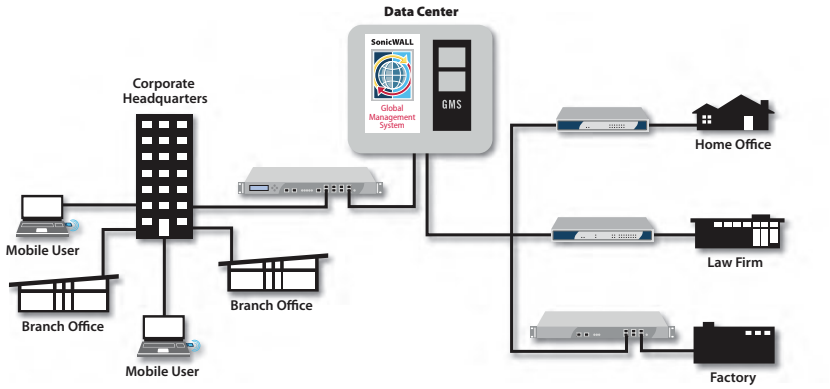
SonicWALL Comprehensive GMS 24x7 Support for 10 Nodes
01-SSC-3353

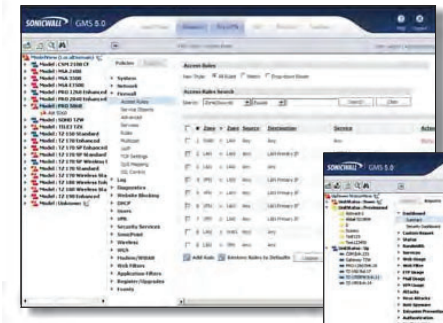
SonicWALL Comprehensive GMS 24x7 Support for 25 Nodes
01-SSC-3374

*Multi-year SKUs also available.

### SonicWALL Global Management System

Providing a comprehensive security management solution for enterprises and service providers.



SonicWALL GMS allows administrators to easily create security policies for the SonicWALL Network Security appliances and enforce them at the global, group or unit level.



SonicWALL GMS allows administrators to generate a wide range of informative and historical reports to provide insight into usage trends, such as which Web sites have been accessed, by whom and security events of the managed SonicWALL Network Security appliances.

### Minimum System Requirement

Below are the minimum requirements for SonicWALL GMS with respect to the operating systems, databases, drivers, hardware and SonicWALL supported appliances:

**Operating System**

Windows 2000 Server (SP4), Windows 2000 Professional (SP4), Windows XP Professional (SP2), Windows 2003 Server (SP2)

**Hardware for Single Deployment**

x86 Environment: Minimum 3 GHz processor Server dual-core CPU Intel processor, 2 GB RAM and 300 GB disk space

**Hardware for Distributed Server Deployment**

| | |
|---|---|
| GMS Server | x86 Environment: Minimum 3 GHz processor single-CPU Intel processor, 2 GB RAM and 300 GB disk space |
| Database Server | x86 Environment: Minimum 3 GHz processor dual-core CPU Intel processor, 2 GB RAM and 300 GB disk space |

**GMS Gateway**

SonicWALL PRO Series Network Security Appliance with minimum firmware, SonicOS Enhanced 3.2 and SonicWALL VPN-based Network Security appliances[1]

**Database**

Microsoft® Environment: Microsoft SQL Server 2000 (SP4) and Microsoft SQL Server 2005 (SP2)[2] on either Windows 2000 Server (SP4) or 2003 Server (SP2)

**Java**

Java Plug-in version 1.5 or later

**Supported SonicWALL Appliances Managed by GMS**

SonicWALL Network Security appliances: E-Class NSA, NSA, PRO Series, TZ Series appliances

SonicWALL Continuous Data Protection appliances

SonicWALL CSM appliances, SonicWALL SSL-VPN appliances

All TCP/IP and SNMP-enabled devices and applications for active monitoring

**Internet Browsers**

Microsoft® Internet Explorer 6.0 or higher

Mozilla Firefox 2.0 or higher

**Supported Firmware**

SonicWALL Network Security appliances: E-Class NSA and NSA SonicOS Enhanced 5.0 or higher

SonicWALL PRO Series appliances: SonicOS Enhanced 3.2 or higher

SonicWALL TZ Series appliances: SonicOS Standard 3.1 or higher and SonicOS Enhanced 3.2 or higher

SonicWALL CDP appliances: SonicWALL CDP 2.3 or higher

SonicWALL CSM appliances: SonicWALL 2.0 or higher

SonicWALL SSL VPN appliances: SonicWALL SSL VPN SMB Firmware 2.0 or higher and SonicWALL Aventail E-Class SSL VPN Firmware 9.0 or higher

[1] The GMS Gateway should have at minimum a SonicWALL PRO 2040 Network Security appliance or higher.

[2] MySQL database is bundled with the GMS application.

**SonicWALL, Inc.**

1143 Borregas Avenue
Sunnyvale CA 94089-1306

T +1 408.745.9600
F +1 408.745.9300

www.sonicwall.com

SONICWALL®
PROTECTION AT THE SPEED OF BUSINESS®

©2008 SonicWALL, Inc. SonicWALL and Protection at the Speed of Business are registered trademarks of SonicWALL, Inc. Other product names mentioned herein may be trademarks and/or registered trademarks of their respective companies. Specifications and descriptions subject to change without notice. 07/08 SW 370

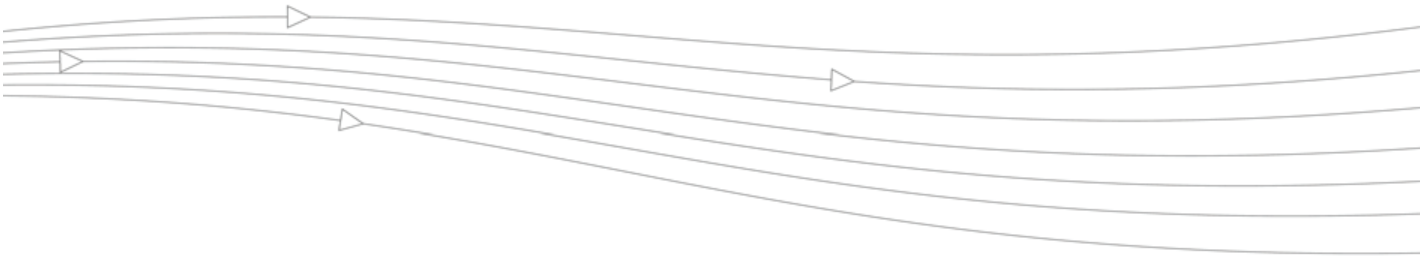
https://web.archive.org/web/20100821022839/http://www.sonicwall.com:80/downloads/SonicWALL_ViewPoint_6_0_Admin_Guide.pdf

**SonicWALL**
**Reporting Solutions**

REPORTING | **SonicWALL ViewPoint**

# SonicWALL ViewPoint 6.0 Administrator's Guide

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1239
**Keysight, EX-1045**
**Page 1239**



# SonicWALL ViewPoint Administrator's Guide

Version 6.0

**SonicWALL, Inc.**
2001 Logic Drive
San Jose, CA 95124-3452
Phone: +1.408.745.9600
Fax: +1.408.745.9300
E-mail: info@sonicwall.com



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1240
Keysight, EX-1045
Page 1240

## Copyright Notice

© 2010 SonicWALL, Inc.

All rights reserved.

Under the copyright laws, this manual or the software described within, can not be copied, in whole or part, without the written consent of the manufacturer, except in the normal use of the software to make a backup copy. The same proprietary and copyright notices must be affixed to any permitted copies as were affixed to the original. This exception does not allow copies to be made for others, whether or not sold, but all of the material purchased (with all backup copies) can be sold, given, or loaned to another person. Under the law, copying includes translating into another language or format.

Specifications and descriptions subject to change without notice.

## Trademarks

SonicWALL is a registered trademark of SonicWALL, Inc.

Windows XP, Windows Vista, Windows 7, Windows Server 2008, Windows Server 2003, Internet Explorer, and Active Directory are trademarks or registered trademarks of Microsoft Corporation.

Firefox is a trademark of the Mozilla Foundation.

Netscape is a registered trademark of Netscape Communications Corporation in the U.S. and other countries. Netscape Navigator and Netscape Communicator are also trademarks of Netscape Communications Corporation and may be registered outside the U.S.

Adobe, Acrobat, and Acrobat Reader are either registered trademarks or trademarks of Adobe Systems Incorporated in the U.S. and/or other countries.

Other product and company names mentioned herein may be trademarks and/or registered trademarks of their respective companies and are the sole property of their respective manufacturers.

## End User Licensing Agreement For SonicWall ViewPoint

This End User Licensing Agreement (EULA) is a legal agreement between you and SonicWALL, Inc. (SonicWALL) for the SonicWALL software product identified above, which includes computer software and any and all associated media, printed materials, and online or electronic documentation (SOFTWARE PRODUCT). By opening the sealed package(s), installing, or otherwise using the SOFTWARE PRODUCT, you agree to be bound by the terms of this EULA. If you do not agree to the terms of this EULA, do not open the sealed package(s), install or use the SOFTWARE PRODUCT. You may however return the unopened SOFTWARE PRODUCT to your place of purchase for a full refund.

The SOFTWARE PRODUCT is licensed, not sold.

You acknowledge and agree that all right, title, and interest in and to the SOFTWARE PRODUCT, including all associated intellectual property rights, are and shall remain with SonicWALL. This EULA does not convey to you an interest in or to the SOFTWARE PRODUCT, but only a limited right of use revocable in accordance with the terms of this EULA.

- The SOFTWARE PRODUCT is licensed as a single product and can only be used as such.
- You may also store or install a copy of the SOFTWARE PRODUCT on a storage device, such as a network server, used only to install or run the SOFTWARE PRODUCT on your other computers over an internal network.
- You may not resell, or otherwise transfer for value, rent, lease, or lend the SOFTWARE PRODUCT.
- The SOFTWARE PRODUCT is trade secret or confidential information of SonicWALL or its licensors. You shall take appropriate action to protect the confidentiality of the SOFTWARE PRODUCT. You shall not reverse-engineer, de-compile, or disassemble the SOFTWARE PRODUCT, in whole or in part. The provisions of this section will survive the termination of this EULA.
- You agree and certify that neither the SOFTWARE PRODUCT nor any other technical data received from SonicWALL, nor the direct product thereof, will be exported outside the United States except as permitted by



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1241
Keysight, EX-1045
Page 1241

the laws and regulations of the United States, which may require U.S. Government export approval/licensing. Failure to strictly comply with this provision shall automatically invalidate this License.

## License

SonicWALL grants you a non-exclusive license to use the SOFTWARE PRODUCT for a number of SonicWALL eligible products. This number is specified and shipped with the SOFTWARE PRODUCT. Support for additional SonicWALL eligible products is subject to a separate upgrade license.

## Upgrades

If the SOFTWARE PRODUCT is labeled as an upgrade, you must be properly licensed to use a product identified by SonicWALL as being eligible for the upgrade in order to use the SOFTWARE PRODUCT. A SOFTWARE PRODUCT labeled as an upgrade replaces and/or supplements the product that formed the basis for your eligibility for the upgrade. You may use the resulting upgraded product only in accordance with the terms of this EULA. If the SOFTWARE PRODUCT is an upgrade of a component of a package of software programs that you licensed as a single product, the SOFTWARE PRODUCT may be used and transferred only as part of that single product package and may not be separated for use on more than one computer.

## Support Services

SonicWALL may provide you with support services related to the SOFTWARE PRODUCT (“Support Services”). Use of Support Services is governed by the SonicWALL policies and programs described in the user manual, in “online” documentation, and/or in other SonicWALL-provided materials. Any supplemental software code provided to you as part of the Support Services shall be considered part of the SOFTWARE PRODUCT and subject to terms and conditions of this EULA. With respect to technical information you provide to SonicWALL as part of the Support Services, SonicWALL may use such information for its business purposes, including for product support and development. SonicWALL shall not utilize such technical information in a form that identifies its source.

## Ownership

As between the parties, SonicWALL retains all title to, ownership of, and all proprietary rights with respect to the SOFTWARE PRODUCT (including but not limited to any images, photographs, animations, video, audio, music, text, and ‘applets” incorporated into the SOFTWARE PRODUCT), the accompanying printed materials, and any copies of the SOFTWARE PRODUCT. The SOFTWARE PRODUCT is protected by copyrights laws and international treaty provisions. The SOFTWARE PRODUCT is licensed, not sold. This EULA does not convey to you an interest in or to the SOFTWARE PRODUCT, but only a limited right of use revocable in accordance with the terms of this EULA.

## U.S. Government Restricted Rights

If you are acquiring the Software including accompanying documentation on behalf of the U.S. Government, the following provisions apply. If the Software is supplied to the Department of Defense (“DoD”), the Software is subject to “Restricted Rights”, as that term is defined in the DOD Supplement to the Federal Acquisition Regulations (“DFAR”) in paragraph 252.227 7013(c) (1). If the Software is supplied to any unit or agency of the United States Government other than DOD, the Government’s rights in the Software will be as defined in paragraph 52.227 19(c) (2) of the Federal Acquisition Regulations (“FAR”). Use, duplication, reproduction or disclosure by the Government is subject to such restrictions or successor provisions. Contractor/Manufacturer is: SonicWALL, Inc. 2001 Logic Drive, San Jose, CA 95124-3452, USA.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1242
Keysight, EX-1045
Page 1242

## Exports License

Licensee will comply with, and will, at SonicWALL's request, demonstrate such compliance with all applicable export laws, restrictions, and regulations of the U.S. Department of Commerce, the U.S. Department of Treasury and any other any U.S. or foreign agency or authority. Licensee will not export or re-export, or allow the export or re-export of any product, technology or information it obtains or learns pursuant to this Agreement (or any direct product thereof) in violation of any such law, restriction or regulation, including, without limitation, export or re-export to Cuba, Iran, Iraq, Libya, North Korea, Sudan, Syria or any other country subject to applicable U.S. trade embargoes or restrictions, or to any party on the U.S. Export Administration Table of Denial Orders or the U.S. Department of Treasury List of Specially Designated Nationals, or to any other prohibited destination or person pursuant to U.S. law, regulations or other provisions.

## Miscellaneous

This EULA represents the entire agreement concerning the subject matter hereof between the parties and supercedes all prior agreements and representations between them. It may be amended only in writing executed by both parties. This EULA shall be governed by and construed under the laws of the State of California as if entirely performed within the State and without regard for conflicts of laws. Should any term of this EULA be declared void or unenforceable by any court of competent jurisdiction, such declaration shall have no effect on the remaining terms hereof. The failure of either party to enforce any rights granted hereunder or to take action against the other party in the event of any breach hereunder shall not be deemed a waiver by that party as to subsequent enforcement of rights or subsequent actions in the event of future breaches.

## Termination

This EULA is effective upon your opening of the sealed package(s), installing or otherwise using the SOFTWARE PRODUCT, and shall continue until terminated. Without prejudice to any other rights, SonicWALL may terminate this EULA if you fail to comply with the terms and conditions of this EULA. SonicWALL reserves the right to terminate this EULA five (5) years after the SOFTWARE PRODUCT is issued to Licensee. In event of termination, you agree to return or destroy the SOFTWARE PRODUCT (including all related documents and components items as defined above) and any and all copies of same.

## Limited Warranty

SonicWALL warrants that a) the software product will perform substantially in accordance with the accompanying written materials for a period of ninety (90) days from the date of purchase, and b) any support services provided by SonicWALL shall be substantially as described in applicable written materials provided to you by SonicWALL. Any implied warranties on the software product are limited to ninety (90) days. Some states and jurisdictions do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you.

## Customer Remedies

SonicWALL's and its suppliers' entire liability and your exclusive remedy shall be, at SonicWALL's option, either a) return of the price paid, or b) repair or replacement of the SOFTWARE PRODUCT that does not meet SonicWALL's Limited Warranty and which is returned to SonicWALL with a copy of your receipt. This Limited Warranty is void if failure of the SOFTWARE PRODUCT has resulted from accident, abuse, or misapplication. Any replacement SOFTWARE PRODUCT shall be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer. Outside of the United States, neither these remedies nor any product Support Services offered by SonicWALL are available without proof of purchase from an authorized SonicWALL international reseller or distributor.

## No Other Warranties

To the maximum extent permitted by applicable law, SonicWALL and its suppliers/licensors disclaim all other warranties and conditions, either express or implied, including, but not limited to, implied warranties of merchantability, fitness for a particular purpose, title, and non-infringement, with regard to the SOFTWARE PRODUCT, and the provision of or failure to provide support services. This limited warranty gives you specific legal rights. You may have others, which vary from state/jurisdiction to state/jurisdiction.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1243
Keysight, EX-1045
Page 1243

## Limitation of Liability

Except for the warranties provided hereunder, to the maximum extent permitted by applicable law, in no event shall SonicWALL or its suppliers/licensors be liable for any special, incidental, indirect, or consequential damages for lost business profits, business interruption, loss of business information,) arising out of the use of or inability to use the SOFTWARE PRODUCT or the provision of or failure to provide support services, even if SonicWALL has been advised of the possibility of such damages. In any case, SonicWALL's entire liability under any provision of this EULA shall be limited to the amount actually paid by you for the SOFTWARE PRODUCT; provided, however, if you have entered into a SonicWALL support services agreement, SonicWALL's entire liability regarding support services shall be governed by the terms of that agreement. Because some states and jurisdiction do not allow the exclusion or limitation of liability, the above limitation may not apply to you.

Manufacturer is SonicWALL, Inc. with headquarters located at 2001 Logic Drive, San Jose, CA 95124-3452, USA.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1244
Keysight, EX-1045
Page 1244

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1245
Keysight, EX-1045
Page 1245

# Table of Contents


**Table of Contents ..... 1**

**Chapter 1: Introduction to SonicWALL ViewPoint ..... 9**

SonicWALL ViewPoint Overview ..... 9
SonicWALL ViewPoint Installation ..... 10
License and Registration Requirements ..... 10
Accessing the Correct Management Interface ..... 11
Switching Between Management Interfaces ..... 12
Tips and Tutorials ..... 12
Navigating the ViewPoint User Interface ..... 13
UTM Panel ..... 13
SSL-VPN Panel ..... 15
Console Panel ..... 16
ViewPoint Views and Status ..... 17
Using the ViewPoint TreeControl Menu ..... 20
About Signed Applets in SonicWALL ViewPoint ..... 21

**Chapter 2: Using the UMH System Interface ..... 23**

Overview of the UMH System Interface ..... 24
Switching to the Application Interface ..... 24
Viewing Online Help and Tips ..... 24
Logging Out of the UMH System Interface ..... 25
Configuring UMH System Settings ..... 25
Viewing System Status ..... 26
Managing System Licenses ..... 26
Configuring System Administration Settings ..... 28
Managing System Settings ..... 29
Using System Diagnostics ..... 30
Configuring UMH Deployment Options ..... 31





CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1246
Keysight, EX-1045
Page 1246

Configuring the Deployment Role ..... 32
Configuring Deployment Settings ..... 34
Controlling Deployment Services ..... 36

**Chapter 3: Adding SonicWALL Appliances ..... 37**

Adding SonicWALL Appliances to SonicWALL ViewPoint ..... 37
Adding SonicWALL Appliances ..... 38
Modifying SonicWALL Appliance Settings ..... 39
Deleting SonicWALL Appliances from ViewPoint ..... 39

**Chapter 4: Using the SonicToday Panel ..... 41**

Overview of the SonicToday Panel ..... 42
Editing a Component Window ..... 42
Adding a Component Window ..... 44
Application Widget ..... 44
RSS Feed ..... 46
Adding More Pages ..... 47
Editing and Deleting Pages ..... 48
Other Features ..... 49

**Chapter 5: Configuring User Settings ..... 51**

Configuring General Settings ..... 51
Configuring Reports Settings ..... 53
Adding Web Sites to the Filter List ..... 54
Deleting Web Sites from the Filter List ..... 54
Adding Web Users to the Filter List ..... 54
Deleting Web Sites from the Filter List ..... 55

**Chapter 6: Configuring Log Settings ..... 57**

Configuration ..... 57
View Log ..... 58

**Chapter 7: Configuring Management Settings ..... 61**

Settings ..... 61
Configuring Email Settings ..... 62
Configuring System Debug Level ..... 62
Enforcing Password Security ..... 63
Synchronizing Model Codes ..... 63
Alert Settings ..... 64
Sessions ..... 65
Managing Sessions ..... 65



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1247
Keysight, EX-1045
Page 1247

Database Maintenance ..... 66
Configuring Backup Schedule and Settings ..... 67
Backing Up a Database Immediately ..... 68
Restoring a Database Backup ..... 68

**Chapter 8: Managing Reports in the Console Panel .....71**

Settings ..... 71
Enabling Report Table Sorting ..... 72
Controlling the Number of Appliances with Log Viewer Enabled ..... 72
Summarizer ..... 73
About Summary Data in Reports ..... 73
Summarizer Settings and Summarization Interval ..... 73
Configuring the Syslog Deletion Schedule Settings ..... 78
Configuring Host Name Resolution ..... 79
Email/Archive ..... 81
Configuring Email/Archive Settings ..... 81
Scheduled Reports ..... 82
Management ..... 87
Configuring Report Data Management ..... 87

**Chapter 9: Using Diagnostics .....89**

Summarizer Status ..... 90

**Chapter 10: Granular Event Management .....97**

Granular Event Management Overview ..... 97
What is Granular Event Management? ..... 98
How Does Granular Event Management Work? ..... 98
Using Granular Event Management ..... 99
About Alerts ..... 100
Configuring Granular Event Management ..... 101
Configuring Events on the Console Panel ..... 101
Enabling or Disabling Alerts on the UTM Panel ..... 107
Viewing Current Alerts ..... 108

**Chapter 11: Web Services .....109**

URI Basics ..... 110
Settings ..... 111
Status ..... 112

**Chapter 12: Using ViewPoint Help .....113**

Tips and Tutorials ..... 113



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1248
Keysight, EX-1045
Page 1248

About ViewPoint ..... 114

**Chapter 13: ViewPoint Reporting Features ..... 115**

ViewPoint Reporting Overview ..... 115
Viewing ViewPoint Reports ..... 117
Navigating ViewPoint Reporting ..... 119
Global Views ..... 120
Unit View ..... 121
Using Interactive Reports ..... 122
Searching for a Report ..... 123
Collapsible TreeControl Pane ..... 128
Enabling/Disabling Scheduled Reports ..... 128
Combined Reports ..... 128
Improved Navigation ..... 129
Showing Domain Names in Reports ..... 130
Managing ViewPoint Reports on the Console Panel ..... 131

**Chapter 14: Scheduling and Configuring Reports ..... 133**

Configuring Scheduled Reports ..... 134
Viewing or Managing Scheduled Reports ..... 134
Adding or Editing a Scheduled Report ..... 135
Selecting Reports for Summarization ..... 137
Configuring Inheritance for Reporting Screens ..... 138
Configuring Data Storage Settings ..... 139
Configuring Summarization Data for Top Usage ..... 140
Configuring Summarization Data for Bandwidth Reports ..... 141
Configuring Dashboard Summary Reports ..... 142
Viewing Current Alerts ..... 144
Scheduling PDF Compliance Reports ..... 144
Compliance Report Overview ..... 144
Adding a New Scheduled Compliance Report ..... 145
Customizing Your Detailed Reports Page ..... 149

**Chapter 15: Viewing Reports ..... 153**

Managing Report Settings ..... 154
Editing Report Settings ..... 154
Selecting a Graphical Display ..... 154
Setting a Date or Date Range ..... 155
Additional Settings ..... 156
Troubleshooting Reports ..... 156
Viewing General Status Reports ..... 157



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1249
Keysight, EX-1045
Page 1249

Viewing Dashboard Reports ..... 159
Viewing the Dashboard Summary Report ..... 159
Viewing the Security Dashboard Report ..... 162
Using Custom Reports on UTM Appliances ..... 163
Toggling Between Split Mode and Full Mode ..... 164
Configuring the Date and Time for Custom Reports ..... 166
Configuring the Report Layout and Generating the Report ..... 168
Generating the Custom Report ..... 176
Viewing a Custom Report ..... 177
Printing a Page or Exporting the Report as a PDF or CSV File ..... 179
Saving the Report Template ..... 180
Viewing Bandwidth Reports ..... 180
Viewing the Bandwidth Summary Report ..... 181
Viewing the Top Users of Bandwidth ..... 183
Viewing Bandwidth Usage Over Time ..... 185
Viewing the Top Users of Bandwidth Over Time ..... 187
Viewing Services Reports ..... 189
Viewing the Services Summary Report ..... 189
Viewing Web Usage Reports ..... 191
Viewing the Web Usage Summary Report ..... 192
Viewing the Top Web Sites ..... 194
Viewing the Top Users of Web Bandwidth ..... 195
Viewing Web Usage by User ..... 197
Viewing Web Usage By Site ..... 199
Viewing Web Usage By Category ..... 200
Viewing Web Usage Over Time ..... 202
Viewing Top Sites Over Time ..... 203
Viewing Top Users Over Time ..... 205
Viewing Web Usage By User Over Time ..... 207
Viewing Web Usage By Category Over Time ..... 208
Viewing Web Filter Reports ..... 209
Viewing the Web Filter Summary Report ..... 210
Viewing the Web Filter Top Sites Report ..... 212
Viewing the Top Users that Try to Access Blocked Sites ..... 213
Viewing the Blocked Sites for Each User ..... 215
Viewing Blocked Sites Sorted By Site ..... 216
Viewing Blocked Sites Sorted By Category ..... 217
Viewing Blocked Site Attempts Over Time ..... 219
Viewing the Top Blocked Site Attempts Over Time ..... 220
Viewing the Top Blocked Site Users Over Time ..... 221
Viewing Blocked Sites for Each User Over Time ..... 222



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1250
Keysight, EX-1045
Page 1250

Viewing Blocked Sites By Category Over Time ....223
Viewing File Transfer Protocol Reports ....225
Viewing the FTP Summary Report ....225
Viewing the Top FTP Sites By User ....227
Viewing FTP Bandwidth Usage Over Time ....228
Viewing the Top Users of FTP Bandwidth Over Time ....230
Viewing Mail Usage Reports ....231
Viewing the Mail Usage Summary Report ....232
Viewing the Top Users of Mail Bandwidth ....234
Viewing Mail Usage Over Time ....235
Viewing the Top Users of Mail Bandwidth Over Time ....237
Viewing VPN Usage Reports ....238
Viewing the VPN Usage Summary Report ....239
Viewing the Top VPN Users ....241
Viewing VPN Usage Over Time ....242
Viewing the Top VPN Users Over Time ....243
Viewing VPN Usage By Policy ....245
Viewing the Top VPN Policies Over Time ....246
Viewing Hourly VPN Usage By Policy ....248
Viewing the VPN Services Summary Report ....249
Viewing Attacks Reports ....250
Viewing the Attack Summary Report ....251
Viewing the Attacks By Category ....253
Viewing the Errors Report ....254
Viewing Attack Reports Over Time ....256
Viewing the Attacks By Category Over Time ....257
Viewing Errors Over Time ....258
Viewing Virus Attacks Reports ....260
Viewing the Top Viruses By Attack Attempts Report ....262
Viewing the Virus Attack Attempts Report ....263
Viewing the Virus Attacks By User Report ....265
Viewing Anti-Spyware Reports ....266
Viewing a Spyware Summary ....268
Viewing Spyware Attempts By Category ....269
Viewing Spyware Attempts Over Time ....270
Viewing Spyware Attempts By Category Over Time ....272
Viewing Intrusion Prevention Reports ....273
Viewing the Intrusion Prevention Summary Report ....275
Viewing Intrusion Attempts By Category ....276
Viewing Intrusions Over Time ....278
Viewing Intrusion Reports By Category Over Time ....280



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1251
Keysight, EX-1045
Page 1251

Viewing Application Firewall Reports ..... 281
Viewing the Application Firewall Summary Report ..... 282
Viewing the Application Firewall Over Time Report ..... 283
Viewing Application Firewall Top Applications ..... 284
Viewing Application Firewall Top Users ..... 285
Viewing Application Firewall Top Policies ..... 286
Viewing Authentication Reports ..... 287
Viewing the User Login Report ..... 288
Viewing the Administrator Login Report ..... 289
Viewing the Failed Login Report ..... 289
Viewing the Log ..... 290
Viewing the Log for a SonicWALL Appliance ..... 291

**Chapter 16: SSL-VPN Reporting ..... 293**

SSL-VPN Reporting Overview ..... 293
What is SSL-VPN Reporting? ..... 294
Benefits of SSL-VPN Reporting ..... 294
How Does SSL-VPN Reporting Work? ..... 295
Using and Configuring SSL-VPN Reporting ..... 295
About Viewing Available SSL-VPN Report Types ..... 295
Configuring SSL-VPN Scheduled Reports ..... 296
Configuring SSL-VPN Summarization ..... 297

**Chapter 17: Viewing SSL-VPN Reports ..... 299**

Viewing General Status Reports ..... 299
Viewing SSL-VPN Bandwidth Reports ..... 301
Viewing SSL-VPN Bandwidth Summary Reports ..... 301
Viewing SSL-VPN Top Users of Bandwidth Reports ..... 303
Viewing SSL-VPN Bandwidth Usage Over Time Reports ..... 304
Viewing SSL-VPN Top Users of Bandwidth Over Time Reports ..... 306
Using SSL-VPN Custom Reports ..... 307
Toggling Between Split Mode and Full Mode ..... 308
Configuring the Date and Time for Custom Reports ..... 311
Configuring the Report Layout and Generating the Report ..... 314
Generating the Custom Report ..... 320
Viewing a Custom Report ..... 321
Printing a Page or Exporting the Report as a PDF or CSV File ..... 323
Saving the Report Template ..... 324
Viewing SSL-VPN Resources Reports ..... 325
Viewing SSL-VPN Resources Summary Reports ..... 325
Viewing SSL-VPN Resources Top Users Reports ..... 327



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1252
Keysight, EX-1045
Page 1252

Viewing SSL-VPN Authentication Reports ....330
Viewing SSL-VPN User Login Reports ....330
Viewing SSL-VPN Failed Login Reports ....331
Viewing the SSL-VPN Log ....332
Viewing the Log for a SSL-VPN Appliance ....332

**Appendix A: Installing SonicWALL ViewPoint ....335**

About Installing and Upgrading SonicWALL ViewPoint ....336
Installing SonicWALL ViewPoint ....336
Installation Overview ....336
Activating SonicWALL ViewPoint on Your Appliances ....340
Registering Your SonicWALL Appliance ....341
Activating the ViewPoint Software on Your Appliance ....341
Enabling the ViewPoint License on Your Appliance ....342
Installing Universal Management Suite ....342
Upgrading SonicWALL ViewPoint 5.1 to 6.0 ....349
Registering SonicWALL ViewPoint ....351
Configuring Deployment Settings ....354
Configuring Web Port Settings ....354
Configuring SMTP Settings ....355
Upgrading from ViewPoint to GMS ....356
Enabling the GMS Free Trial from ViewPoint ....357
Enabling the GMS Free Trial from the UMH Interface ....359
Completing the Free Trial Upgrade ....360
Configuring Appliances for GMS Management ....364
Purchasing a SonicWALL GMS Upgrade ....366
Miscellaneous Procedures and Troubleshooting Tips ....368
Miscellaneous Procedures ....368
Troubleshooting Tips ....370

**Appendix B: Technical Tips ....373**

Log Viewer ....373
Real-time Syslog Viewer ....375
Forwarding Syslog Data to Another Syslog Server ....376
Posting ViewPoint Reporting to Another Web Server for End-User Access ....377

**Index ....379**



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1253
Keysight, EX-1045
Page 1253

# CHAPTER 1
# Introduction to SonicWALL ViewPoint

This chapter provides an overview of SonicWALL ViewPoint and information about the user interface.

See the following sections:


- "SonicWALL ViewPoint Overview" on page 9
- "SonicWALL ViewPoint Installation" on page 10
- "Accessing the Correct Management Interface" on page 11
- "Navigating the ViewPoint User Interface" on page 13
- "ViewPoint Views and Status" on page 17
- "Using the ViewPoint TreeControl Menu" on page 20
- "About Signed Applets in SonicWALL ViewPoint" on page 21


## SonicWALL ViewPoint Overview

Monitoring critical network events and activity, such as security threats, inappropriate Web use, and bandwidth levels, is an essential component of network security. SonicWALL ViewPoint Reporting complements SonicWALL's network security offerings by providing detailed and comprehensive reports of network activity.

TheViewPoint Reporting Module is a software application that creates dynamic, Web-based network reports. The ViewPoint Reporting Module generates both real-time and historical reports to offer a complete view of all activity through SonicWALL network security appliances. With ViewPoint Reporting, you can monitor network access, enhance security, and anticipate future bandwidth needs.




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1254
Keysight, EX-1045
Page 1254

The ViewPoint Reporting Module:

- Displays bandwidth use by IP address and service
- Identifies inappropriate Web use
- Provides detailed reports of attacks
- Collects and aggregates system and network errors
- Shows VPN events and problems
- Presents visitor traffic to your Web site
- Provides detailed daily logs to analyze specific events.

# SonicWALL ViewPoint Installation

SonicWALL ViewPoint can be installed as a fresh install or as an upgrade to SonicWALL ViewPoint 5.0 and above.

Beginning in SonicWALL ViewPoint 5.1, all software components related to SonicWALL ViewPoint and SonicWALL Global Management System (GMS), including the MySQL database, executable binary files for all services, and other necessary files, are installed using the Universal Management Suite (UMS) single-binary installer. All SonicWALL ViewPoint and SonicWALL GMS files are installed as part of the Universal Management Suite, but no distinction is made between SonicWALL ViewPoint and SonicWALL GMS during the installation. The initial installation phase takes just a few minutes for any type of installation, such as a SonicWALL ViewPoint server, a SonicWALL GMS server, a database server, or any other role.

To install the Universal Management Suite from the single binary installer, see the "Installing Universal Management Suite" section on page 342.

## License and Registration Requirements

SonicWALL ViewPoint is registered and licensed from the Windows server on which it is installed. SonicWALL ViewPoint registration is performed using the SonicWALL Universal Management Host system interface.The "Registering SonicWALL ViewPoint" section on page 351 provides detailed instructions for registering and licensing SonicWALL ViewPoint on your Windows system.

On SonicWALL appliances that send reporting data to SonicWALL ViewPoint, ViewPoint is licensed and activated separately from SonicOS. MySonicWALL provides a way to associate SonicWALL appliances with the SonicWALL ViewPoint instance installed on the Windows system. Licensing your SonicWALL ViewPoint application on a SonicWALL appliance requires:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1255
Keysight, EX-1045
Page 1255

- **A MySonicWALL account**. A MySonicWALL account allows you to manage your SonicWALL products and purchase licenses for various services. Creating a MySonicWALL account is fast, simple, and free. Simply complete an online registration form directly from your SonicWALL security appliance management interface. Your MySonicWALL account is also accessible at <https://www.mysonicwall.com> from any Internet connection with a Web browser. Once you have an account, you can purchase SonicWALL ViewPoint and other licenses for your registered SonicWALL security appliances.
- **A registered SonicWALL security appliance with active Internet connection**. You need to register your SonicWALL security appliance to activate SonicWALL ViewPoint. Registering your SonicWALL security appliance is a simple procedure done directly from the management interface. Once your SonicWALL security appliance is registered, you can activate SonicWALL ViewPoint by using an activation key or by synchronizing with mysonicwall.com.

# Accessing the Correct Management Interface

SonicWALL ViewPoint includes two separate management interfaces:

- **SonicWALL Universal Management Host (UMH) System Management Interface** – Used for system management of the SonicWALL ViewPoint instance, including registration and licensing, setting the admin password, creating backups, restarting the system, configuring network settings, selecting the deployment role, and configuring other system settings.
  Access the system management interface with the URL:

  http://<IP address>:<port>/appliance/

  If you are using the standard HTTP port, 80, it is not necessary to append the port number to the IP address. If you are accessing the interface from the same system on which it is installed, use the following URL:

  http://localhost/appliance/

- **SonicWALL ViewPoint Management Interface** – Used to access the SonicWALL ViewPoint application that runs on the system. This interface is used to configure and view SonicWALL ViewPoint reporting on SonicWALL appliances and for configuring SonicWALL ViewPoint administrative settings. Access the SonicWALL ViewPoint management interface with one of the following URLs:

  http://<IP address>:<port>/sgms/
  http://localhost/sgms/

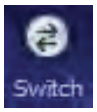


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1256
Keysight, EX-1045
Page 1256

# Switching Between Management Interfaces

You can easily switch between the SonicWALL UMH system management interface and the SonicWALL ViewPoint application management interface.

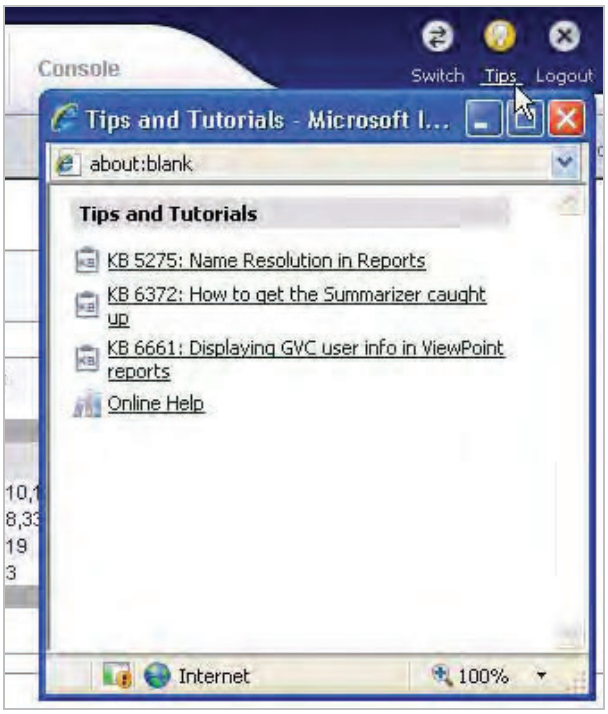
One methods is to change the URL by adding **/sgms** for the ViewPoint application interface or adding **/appliance** for the UMH interface.




A second method involves clicking the **Switch** icon. While logged into either interface, you can switch to the login page of the other interface by clicking the **Switch** button in the top right corner of the page.

# Tips and Tutorials

Tips and tutorials are also available in some section of the user interface, and are denoted by a “Lightbulb” icon:




To access tips and tutorials:

1. Navigate to the page where you need help.
2. If available, click the Lightbulb icon in the upper right-hand corner of the window. Tips, tutorials, and online help are displayed for this topic.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1257
Keysight, EX-1045
Page 1257

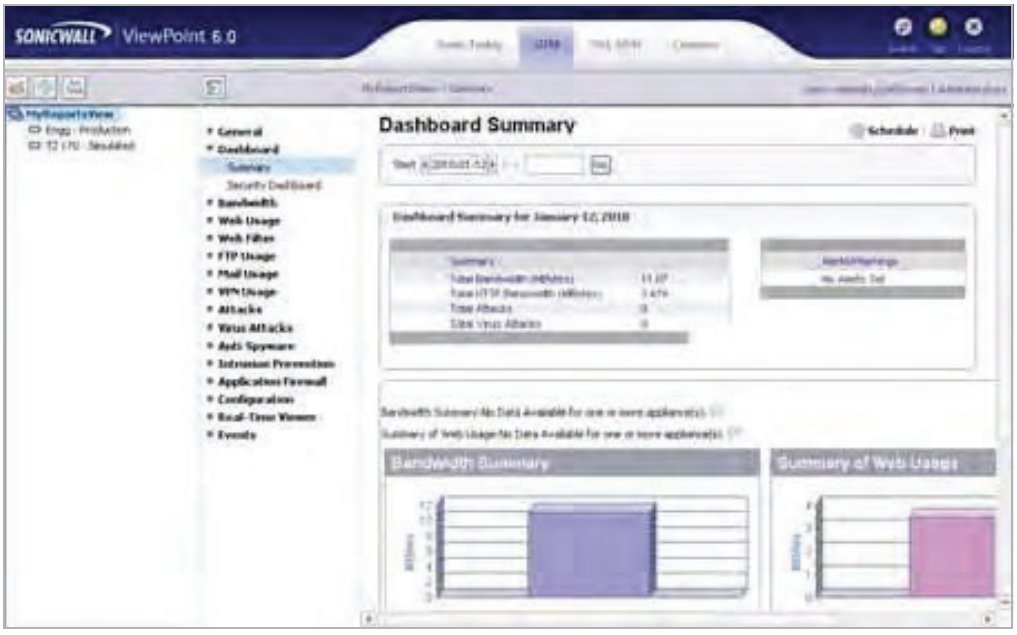
# Navigating the ViewPoint User Interface

This section describes the UTM, SSL-VPN, and Console panels in the SonicWALL ViewPoint user interface. For information about the SonicToday panel, see the Using the SonicToday Panel chapter.

## UTM Panel

The UTM Panel is an essential component of network security that is used to view and schedule reports about critical network events and activity, such as security threats, inappropriate Web use, and bandwidth levels.

To open the UTM Panel, click the **UTM** tab at the top of the ViewPoint user interface.




From the UTM Panel, you can view the following for connected SonicWALL appliances:

- View general unit status, license status, and syslog settings.
- View the SonicWALL security dashboard. Dashboard reports display an overview of bandwidth, uptime, intrusions and attacks, and alerts for connected SonicWALL UTM appliances. The Security Dashboard report provides data about worldwide security threats that can affect your network. The Dashboard also displays data about threats blocked by the SonicWALL security appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1258
Keysight, EX-1045
Page 1258

- View custom reports of Internet activity or Website filtering at the unit level. Custom reports filter raw syslog data and you can specify start and end dates or a date range such as "Week to date". You can filter by user, domain, protocol, traffic, and full URL categories, depending on the type of custom report. The search template can be saved for use again later with the same appliance.
- View general bandwidth usage. These reports include a daily bandwidth summary report, a top users of bandwidth report, and over-time summary and top users reports.
- View a services report. This report includes information about events and usage of protocols and megabytes.
- View Web bandwidth usage. These reports include a daily bandwidth summary report, a top visited sites report, a top users of Web bandwidth report, a report that contains the top sites of each user, and a weekly summary report.
- View the number of attempts that users made to access blocked websites. These reports include a daily summary report, a top blocked sites report, a top users report, a report that contains the top blocked sites of each user, and a weekly summary report.
- View file transfer protocol (FTP) bandwidth usage. These reports include a daily FTP bandwidth summary report, a top users of FTP bandwidth report, and a weekly summary report.
- View mail bandwidth usage. These reports include a daily mail summary report, a top users of mail report, and a weekly summary report.
- View VPN usage. These reports include a daily VPN summary report, a top users of VPN bandwidth report, and a weekly summary report.
- View reports on attempted attacks and errors. The attack reports include a daily attack summary report, an attack by category report, a top sources of attacks report, and a weekly attack summary report. The error reports include a daily error summary report and a weekly error summary report.
- View reports on attempted virus attacks. Virus attacks reports are available for appliances that are licensed for SonicWALL Gateway Anti-Virus. These reports include the most frequent virus attack attempts, virus attacks by top destinations, virus attacks over time, virus attacks over a period of time, and virus attacks by top destinations over time.
- View reports on attempted spyware attacks. Anti-spyware reports are available for appliances that are licensed for SonicWALL Anti-Spyware. These reports include spyware attacks by category, spyware attacks over time, and spyware attacks by category over time.



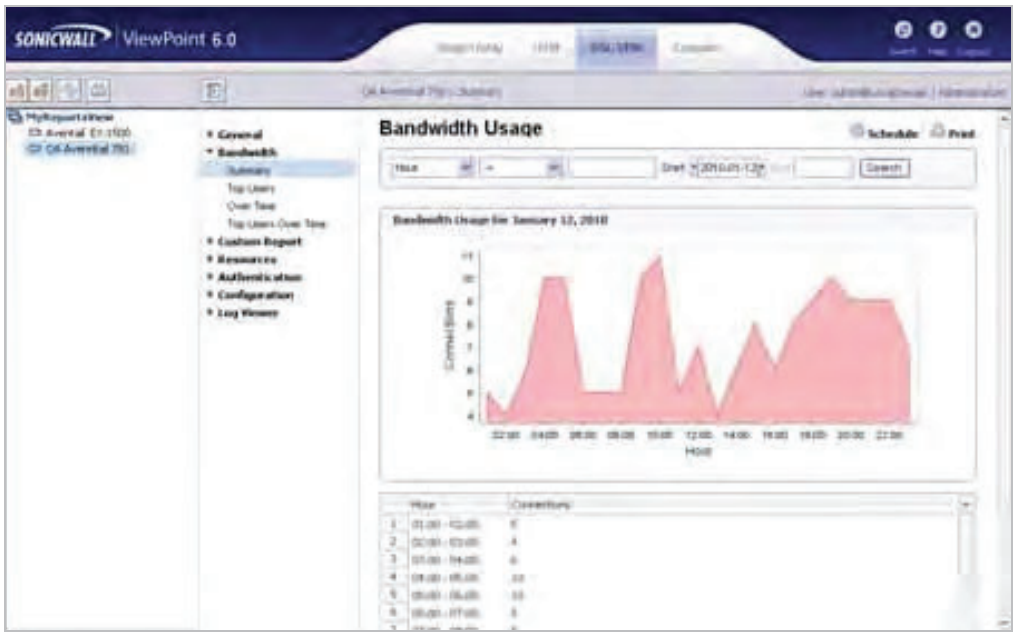
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1259
Keysight, EX-1045
Page 1259

- View reports on attempted intrusion attacks. Intrusion prevention reports are available for appliances that are licensed for SonicWALL Intrusion Prevention Service. These reports include intrusion attacks by source IP address, intrusion attacks by category, intrusion attacks over time, and intrusion attacks by category over time.
- View reports on traffic triggering Application Firewall policies. Application Firewall reports are available for UTM appliances that are licensed for SonicWALL Application Firewall. These reports include summary, over time, top applications, top users, and top policies.
- View successful and unsuccessful user and administrator authentication attempts. These reports include a user authentication report, an administrator authentication report, and a failed authentication report.
- View detailed logging information. The detailed logging information contains each transaction that occurred on the SonicWALL appliance.
- View current alerts and access alert settings.

# SSL-VPN Panel

The SSL-VPN panel provides access to SSL VPN appliances and is similar to the UTM panel. It is used to view and schedule reports about critical network events and activity, such as security threats, inappropriate Web use, and bandwidth levels.

To open the SSL-VPN Panel, click the **SSL-VPN** tab at the top of the ViewPoint user interface.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1260
Keysight, EX-1045
Page 1260

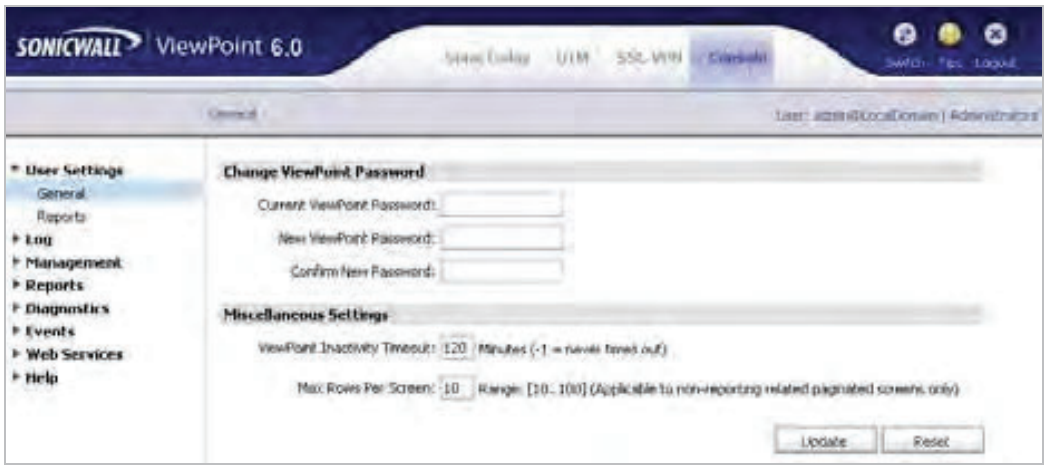
From the SSL-VPN Panel, you can view the following for connected SonicWALL SSL VPN appliances:

- View general unit status, license status, and syslog settings.
- View general bandwidth usage. These reports include a daily bandwidth summary report, a top users of bandwidth report, and over-time summary and top users reports.
- View custom reports of custom reports of resource activity at the unit level. Custom reports filter raw syslog data and you can specify start and end dates or a date range such as "Week to date". You can filter by user, protocol, destination IP, and source IP categories. The search template can be saved for use again later with the same appliance.
- View a resources report. This report includes information about connections and the resource used to connect, such as HTTPS or NetExtender.
- View successful and unsuccessful user authentication attempts. These reports include a user authentication report and a failed authentication report.
- View detailed logging information. The detailed logging information contains each transaction that occurred on the SonicWALL appliance.

# Console Panel

The Console Panel is used to configure SonicWALL ViewPoint settings, view pending tasks, view the log, manage licenses, and configure alerts.

To open the Console Panel, click the **Console** tab at the top of the SonicWALL ViewPoint user interface.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1261
Keysight, EX-1045
Page 1261

From the Console Panel, you can do the following:

- Change the SonicWALL ViewPoint password, adjust the amount of inactive time before the user is automatically logged out of ViewPoint, and set the maximum number of rows displayed on paginated screens.
- Configure Web sites and Web users that will be excluded from Web usage reports.
- View the SonicWALL ViewPoint log and delete old log messages. The SonicWALL ViewPoint log contains information on alert notifications, failed SonicWALL ViewPoint login attempts, and other events that apply to SonicWALL ViewPoint.
- Manage SMTP settings, system email addresses, archive report settings, debug level for logs, and password security settings. You can set the schedule and server settings, and the email alert recipient schedule and preferred format.
- Manage login sessions. You can view the status of user sessions and, if necessary, end them.
- Configure report settings for sort options and maximum units with Log Viewer enabled. Enabling Log Viewer allows custom reports for the system, but is resource intensive.
- Control summarizer settings, syslog and summarized data deletion schedules, and host name resolution settings.
- Configure email archive settings and search settings for scheduled reports, and manage data archiving.
- View summarizer diagnostics, useful for capacity planning.
- Configure granular event management report settings, including threshold, schedule, and alert settings.
- Configure Web services deployment settings and view Web services status.
- View the version number, serial number, and database information for SonicWALL ViewPoint, and access links to all available tips and video tutorials.

# ViewPoint Views and Status

SonicWALL ViewPoint allows you to view status and reports for all appliances at once using **MyReportsView**, or for a single unit at a time with the **Unit** view.

ViewPoint provides status information on the General > Status page of the UTM or SSL-VPN panel.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1262
Keysight, EX-1045
Page 1262

MyReportsView is a grouping of all the appliances you are monitoring with ViewPoint. From the MyReportsView of the UTM or SSL-VPN Panel, Summary and Over Time reports are available for all SonicWALL appliances monitored by SonicWALL ViewPoint.

To open the My Reports view, click the **MyReportsView** icon at the top of the left pane. To display the global status page, navigate to **General > Status**.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1263
Keysight, EX-1045
Page 1263

From the Unit view, reports contain detailed data for the selected SonicWALL appliance. To specify the unit view, click any unit in the left pane. To display the unit status page, navigate to **General > Status** on the **UTM** or **SSL-VPN** panel.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1264
Keysight, EX-1045
Page 1264

# Using the ViewPoint TreeControl Menu

This section describes the content of the TreeControl menu within the SonicWALL ViewPoint user interface.

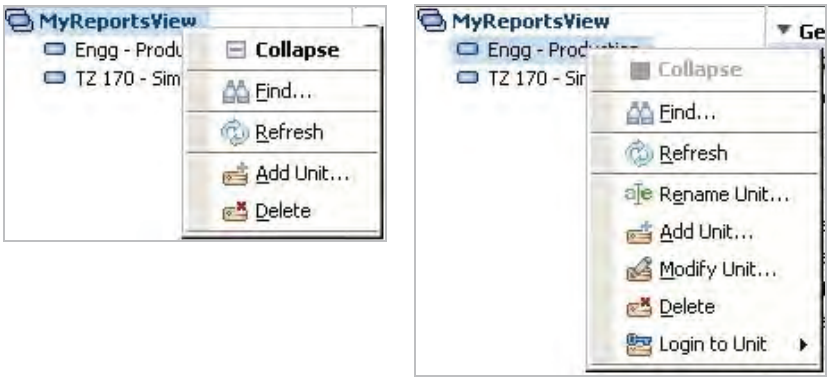
You can control the display of the TreeControl pane by selecting one of the appliance tabs at the top of the main window. For example, when you click the **UTM** tab, the TreeControl pane displays all the connected UTM appliance units. The two appliance tabs can display the following appliance types when ViewPoint is monitoring these device types:

- UTM appliances
- SSL-VPN and EX-Series SRA appliances

You can hide the entire TreeControl pane by clicking the sideways arrow icon, and redisplay the pane by clicking it again. This is helpful when viewing some reports or other extra-wide screens.




To open a TreeControl appliance menu, right-click MyReportsView or a Unit icon.




The following options are available in the right-click menu:

- **Find** – Opens a Find dialog box that allows you to search for units.
- **Refresh** – Refreshes the ViewPoint UI display.
- **Rename Unit** – (unit view only) Renames the selected SonicWALL appliance.
- **Add Unit** – Add a new unit to the ViewPoint view. Requires unit IP and login information.
- **Modify Unit** – (unit view only) Change basic settings for the selected unit, including unit name, IP and login information, and serial number.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1265
Keysight, EX-1045
Page 1265

- **Delete** – Delete the selected unit
- **Login to Unit** – (unit view only) Login to the selected unit using HTTP or HTTPS protocols.

# About Signed Applets in SonicWALL ViewPoint

There are a number of applets in the SonicWALL ViewPoint management interface, such as the TreeControl Applet in the leftmost pane.

Signed Applets refers to a technique for adding a digital signature to a Java applet to prove that it was not tampered with upon receipt from the signer. Signed applets can be given more privileges than ordinary applets. By default, applets have no access to system resources outside the directory from which they were launched, but a signed applet can access local system resources as allowed by the local system's security policy.

In some previous releases of ViewPoint, you were required to edit the java.policy file yourself on the client browser system in order to enable a number of applet related operations, such as Copy/Paste, Import file, Browse local folders, and HTTP/HTTPS login to the managed units from the ViewPoint management interface.

There is no need to edit the java.policy file for signed applets. When a signed applet starts up, a warning pop-up is displayed. If you want to trust the applet, click **Yes**. Copy/paste, Import and HTTP/HTTPS logins will work without any edits to the java.policy file.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1266
Keysight, EX-1045
Page 1266

Otherwise, click **No**. In this case you must manually edit the java.policy file. You can view the following technote for more information about editing the java.policy file:

*Manually Configuring the java.policy File for SonicWALL GMS JRE*



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1267
Keysight, EX-1045
Page 1267

# CHAPTER 2

# Using the UMH System Interface

This chapter describes the Universal Management Host system interface, one of the two management interfaces available for SonicWALL ViewPoint.

This section includes the following subsections:


- Overview of the UMH System Interface, page 24
- Configuring UMH System Settings, page 25
- Configuring UMH Deployment Options, page 31




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1268
Keysight, EX-1045
Page 1268

# Overview of the UMH System Interface

The SonicWALL ViewPoint UMH system interface is used for system management of the SonicWALL ViewPoint instance, including registration and licensing, setting the admin password, configuring network and database settings, selecting the deployment role, and configuring other system settings.

When installing SonicWALL Universal Management Suite 6.0 on a host, a Web server is installed to provide the system management interface. The system interface is available by default at http://localhost/appliance/ after restarting the system.

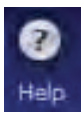



## Switching to the Application Interface




To switch between the System interface and the SonicWALL ViewPoint application interface, click the **Switch** button in the top right corner of the interface.

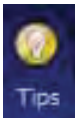
## Viewing Online Help and Tips




To display context sensitive help for the current page, click the **Help** button in the top right corner of the interface.

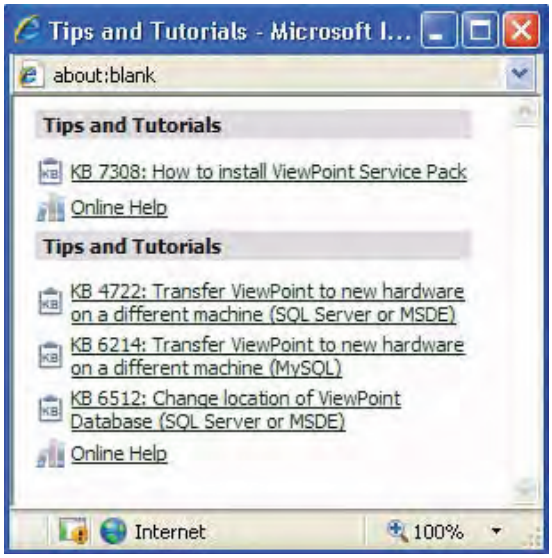


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1269
Keysight, EX-1045
Page 1269

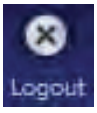



The **Help** button can change to the **Tips** button if the current page has any context sensitive tips or video tutorials.

Clicking on the **Tips** button displays dynamic links for whitepapers, videos, knowledge base articles, other references, and online help.




## Logging Out of the UMH System Interface




To log out of the SonicWALL ViewPoint UMH system interface, click the **Logout** button in the top right corner of the interface.

# Configuring UMH System Settings

This section describes the tasks you can perform on the System pages of the SonicWALL ViewPoint UMH system interface.

See the following sections:

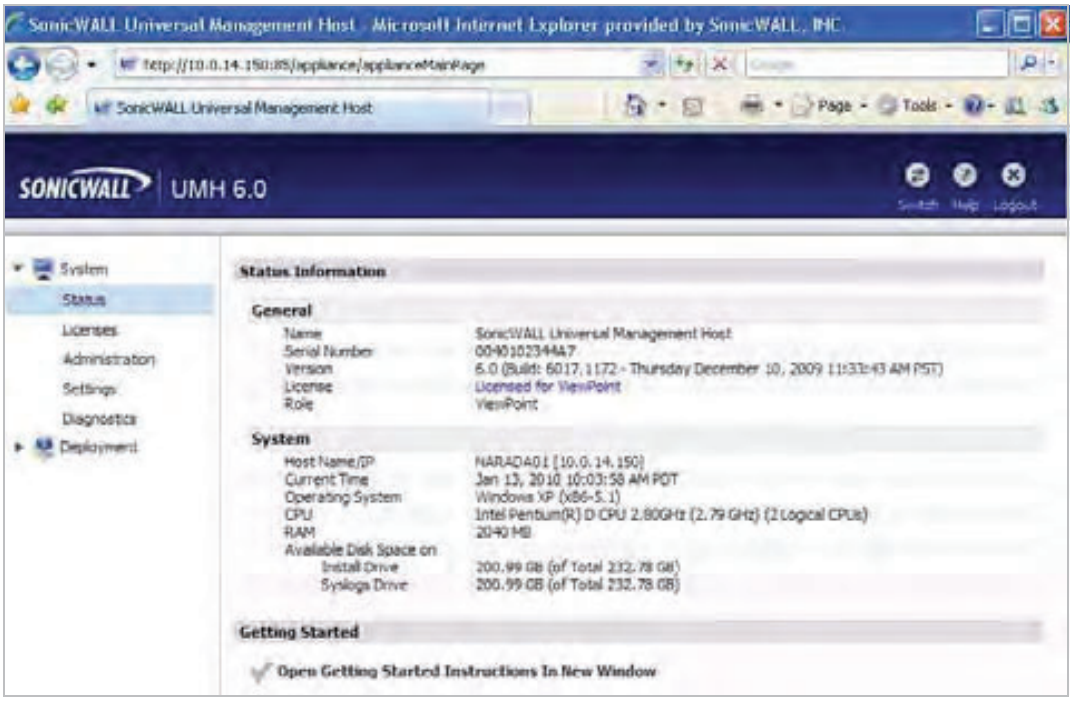

- Viewing System Status, page 26
- Managing System Licenses, page 26
- Configuring System Administration Settings, page 28
- Managing System Settings, page 29
- Using System Diagnostics, page 30




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1270
Keysight, EX-1045
Page 1270

# Viewing System Status

The System > Status page provides the general information about the installation, including the name which identifies the system as a SonicWALL Universal Management Host, the serial number of the SonicWALL ViewPoint instance, the software version, licensing status, and the system role. For SonicWALL ViewPoint, the role is always "ViewPoint."




Under System, the host name of the computer is listed, along with the time and other information about the host computer.

At the bottom of the page, a link is provided to access the *Getting Started Guide* which takes you to the online help table of contents.

# Managing System Licenses

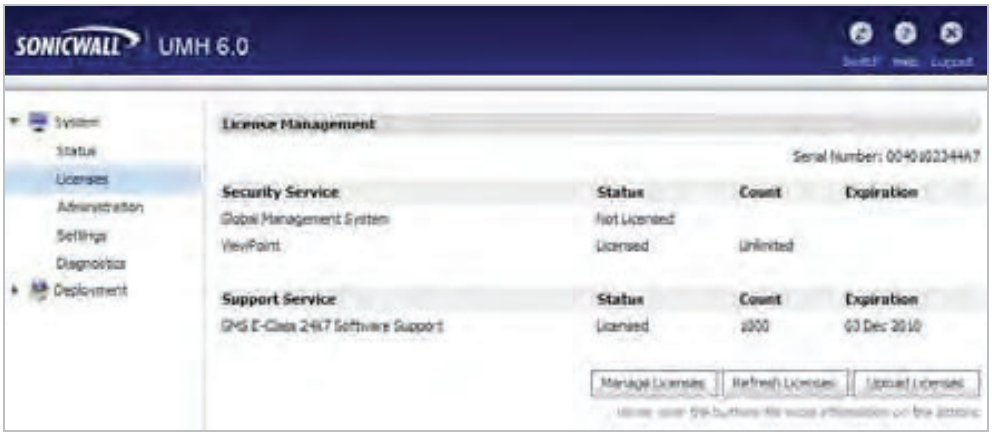
The System > Licenses page provides buttons for managing, refreshing, and uploading licenses. The page displays the status of ViewPoint and Global Management System licenses. The Global Management System license status will show the status of your SonicWALL GMS Free Trial, if activated. If you choose to upgrade to SonicWALL GMS, this page will show Global Management System as fully licensed.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1271
Keysight, EX-1045
Page 1271

The value in the Count column indicates the number of appliances for which this SonicWALL ViewPoint or SonicWALL GMS instance is licensed for reporting or management. For SonicWALL ViewPoint, this value is usually "unlimited", but for SonicWALL GMS, the base license is either for 10 nodes or 25 nodes, and additional node licenses can be purchased in various increments.

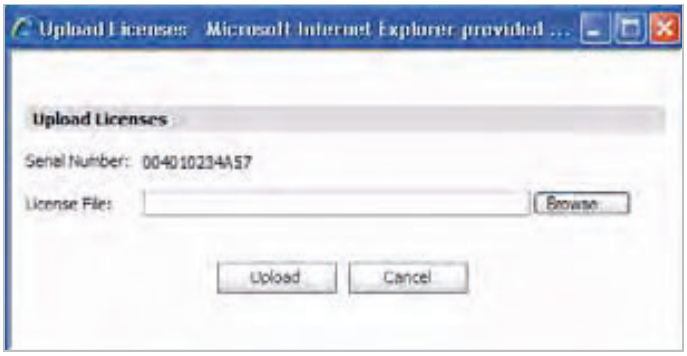
The Expiration column indicates the expiration date of the license. If no date is shown, the license is perpetual, and does not expire.




To display the MySonicWALL login page, click the **Manage Licenses** button. You can purchase licenses and obtain license keysets on MySonicWALL.

Click the **Refresh Licenses** button to refresh the license status on this page.

To upload a new license, click the **Upload Licenses** button and browse to a license file on your computer.

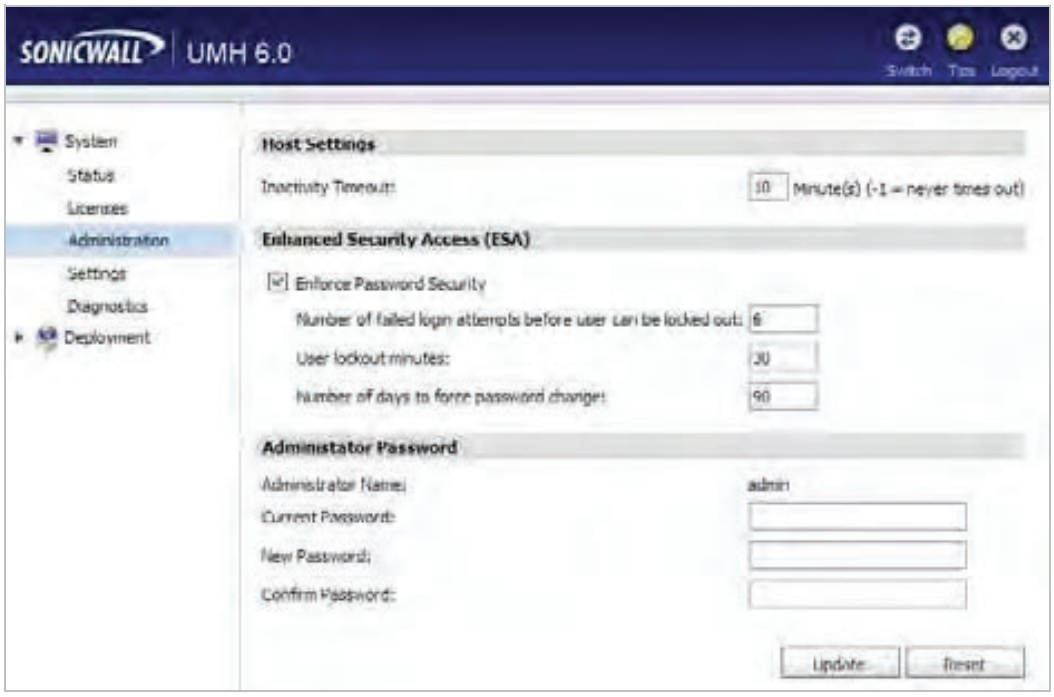





CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1272
Keysight, EX-1045
Page 1272

# Configuring System Administration Settings

The System > Administration page allows you to configure the system behavior for admin login sessions.




Under Host Settings, enter the number of minutes of inactivity allowed before the session is logged out. A setting of **-1** allows an unlimited amount of inactivity without being logged out.

Under Enhanced Security Access, you can configure the number of failed login attempts before the admin account is locked out, and the number of minutes that the lockout lasts. You can also configure the number of days before the admin account password must be changed.

Under Administrator Password, you can change the administrator password for the SonicWALL ViewPoint application. Enter the current password for the system administrator (or root) account into the Current Password field, and then enter the new password into both the **New Password** and **Confirm Password** fields.

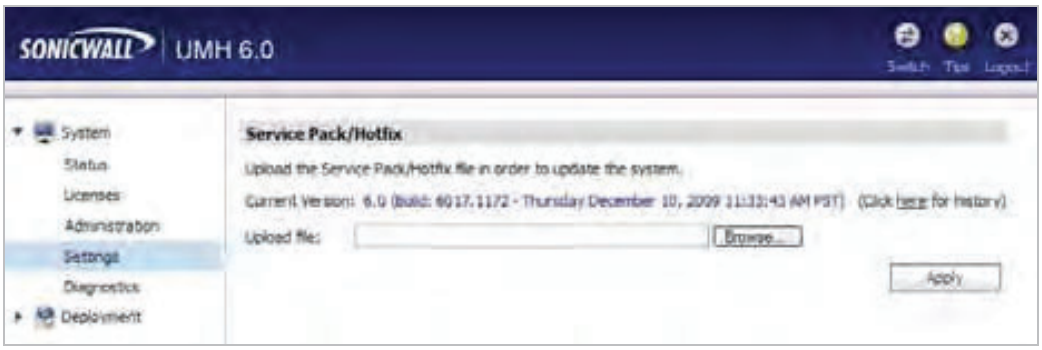
After making any changes on this page, click **Update**. To revert the fields on the page to their default settings, click **Reset**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1273
Keysight, EX-1045
Page 1273

# Managing System Settings

The System > Settings page provides a way to upload new SonicWALL ViewPoint software or service packs to the system. Click **Browse** to browse to the file you wish to upload, and then click **Apply**.



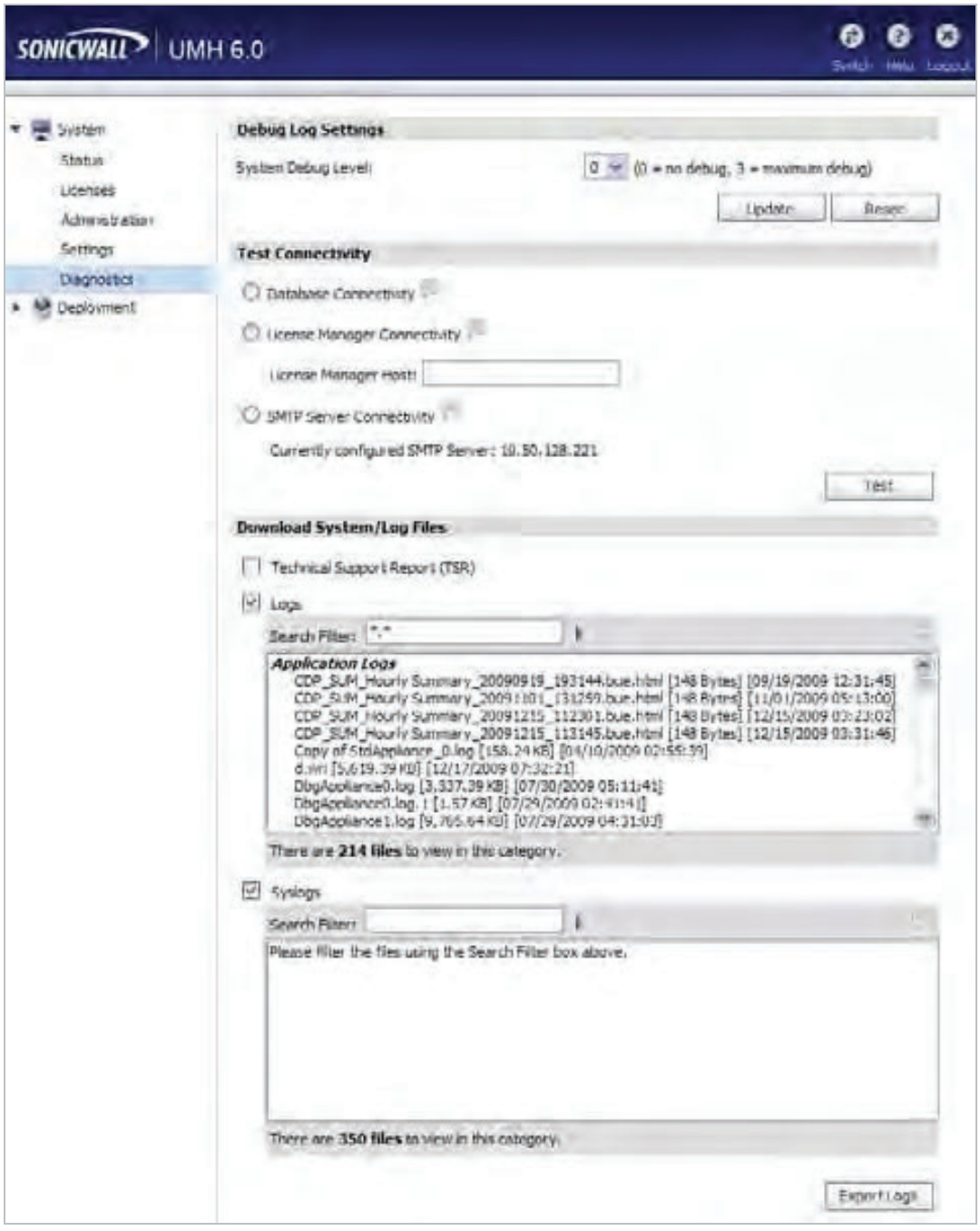

The page shows the current version of SonicWALL UMS, and provides a History link that displays the history of all hotfixes and firmware updates that were applied to the system.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1274
Keysight, EX-1045
Page 1274

# Using System Diagnostics

The System > Diagnostics page is used to set log levels, test connectivity to servers, generate Tech Support Reports, and to search and download system log files.




Under Debug Log Settings, select the log level from the **System Debug Level** drop-down list. You can select **0** for no debug information, **1** or **2** for more, and **3** for maximum debug information.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1275
Keysight, EX-1045
Page 1275

In the Test Connectivity section, select one of the following radio buttons and then click **Test** to verify connectivity to that server:

- **Database Connectivity** – Tests connectivity to the database server configured on the Deployment > Roles page.
- **License Manager Connectivity** – Type the host name or IP address into the License Manager Host field and click Test to test connectivity to that server.
- **SMTP Server Connectivity** – Tests connectivity to the SMTP server configured on the Deployment > Settings page.

In the Download System/Log Files section, you can enter a filter, or search value, into either of the **Search Filter** fields, and then press **Enter**, to locate log entries of interest. Click the **Export Logs** button to save the log files to a file on your computer.

To generate a TSR (Technical Support Report), select the **Technical Support Report (TSR)** checkbox, and then click **Export Logs**.

# Configuring UMH Deployment Options

This section describes the tasks you can perform on the Deployment pages of the SonicWALL ViewPoint UMH system interface.

See the following sections:

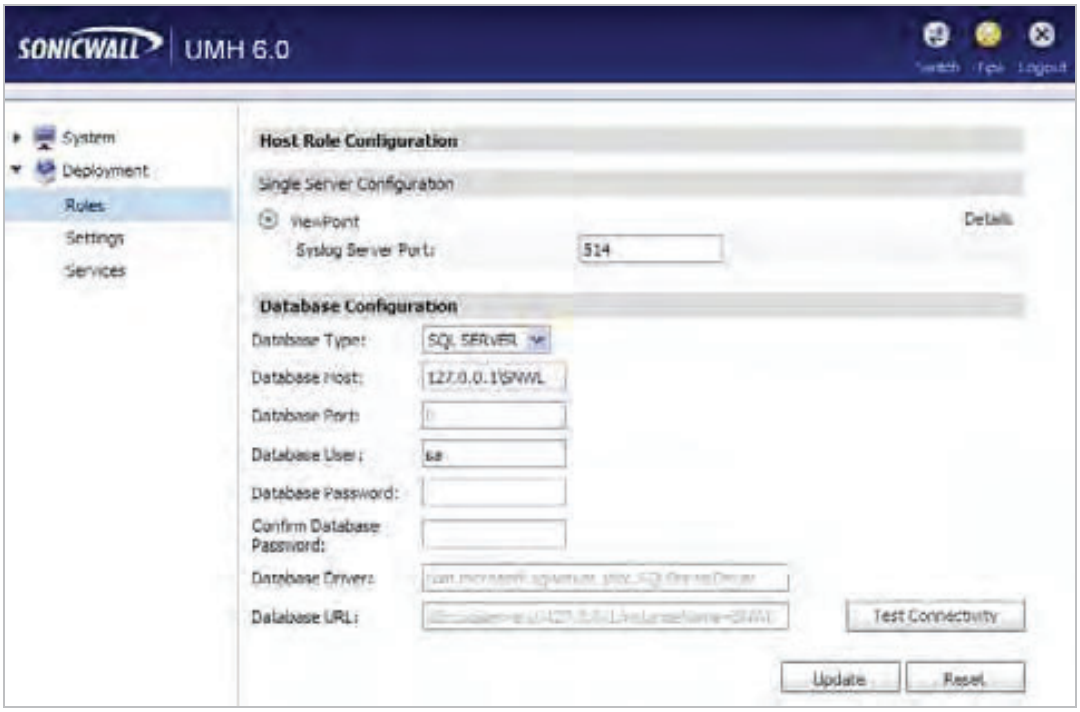

- Configuring the Deployment Role, page 32
- Configuring Deployment Settings, page 34
- Controlling Deployment Services, page 36




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1276
Keysight, EX-1045
Page 1276

# Configuring the Deployment Role

In a SonicWALL ViewPoint installation, the Deployment > Roles page provides a way to configure the syslog port and the database settings, and to test database cnnectivity.




To set the syslog port, enter the port number into the **Syslog Server Port** field.

Under Database Configuration, to provide credentials with which SonicWALL ViewPoint will access the database, enter the account user name into the **Database User** field, and enter the account password into both the **Database Password** and **Confirm Database Password** fields.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1277
Keysight, EX-1045
Page 1277

To test connectivity to the database server, click **Test Connectivity**. A popup will display the status.




When finished, click **Update** to apply the changes. To revert the fields on the page to their default settings, click **Reset**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1278
Keysight, EX-1045
Page 1278

# Configuring Deployment Settings

The Deployment > Settings page provides a way to set the Web ports, the SMTP server IP address with the sender and administrator email addresses, and the SSL access configuration.




To configure the Web ports, enter the desired port numbers into the **HTTP Port** and **HTTPS Port** fields, and then click **Update**.

To configure the SMTP settings, perform the following steps:

1. In the **SMTP Server** field, enter the IP address or fully qualified domain name of the SMTP server. This is normally the same server that handles your regular email service.
2. In the **Sender Address** field, enter the email address, including domain, by which SonicWALL ViewPoint will be known when sending email.
3. In the **Administrator Address** field, enter the email address of the administrator who will receive email alerts and other email communications from SonicWALL ViewPoint.
4. Under SSL Access Configuration, select one of the following settings:

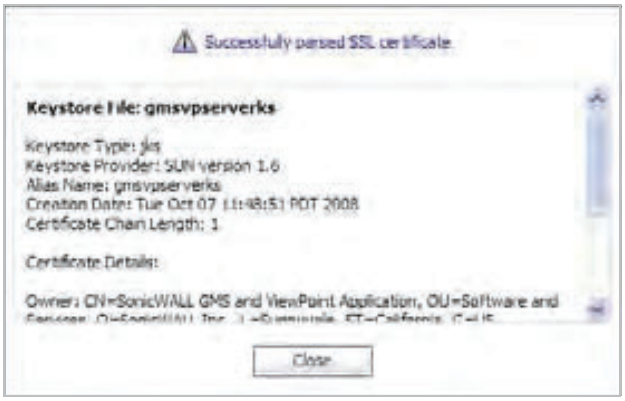


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1279
Keysight, EX-1045
Page 1279

- **Default** – Keep the default certificate that comes with the application for use by the ViewPoint Web Server for SSL access. The filename for the keystore is **gmsvpserverks**.
- **Custom** – Upload a custom certificate for use by the ViewPoint Web Server for SSL access. The original filename of the imported certificate is replaced with **gmsvpservercustomks** in the local file system.

  Click **Browse** and select the certificate file for the **Keystore/Certificate file** field and type the password into the **Keystore/Certificate password** field.

  To display information contained in the certificate, click **View**.



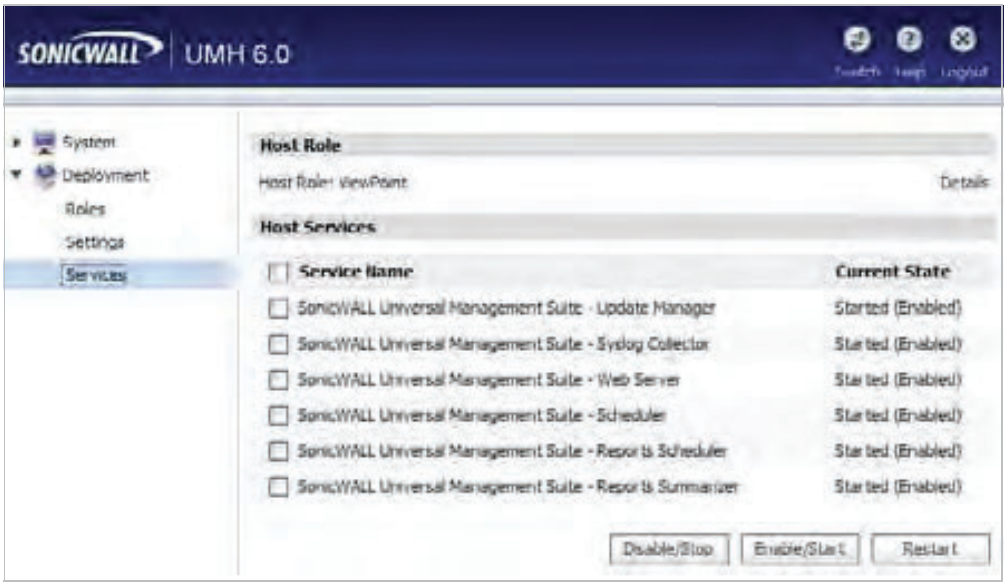

5. When finished, click **Update** to apply the changes. To revert the fields on the page to their default settings, click **Reset**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1280
Keysight, EX-1045
Page 1280

# Controlling Deployment Services

The Deployment > Services page provides a list of the services that are running on your system as part of SonicWALL ViewPoint. It also provides a way to stop or start any of the services.




To stop a service that is currently Enabled, select the checkbox for that service and then click **Disable/Stop**.

To start a service that is currently Disabled, select the checkbox for that service and then click **Enable/Start**.

To restart a service that is either Enabled or Disabled, select the checkbox for that service and then click **Restart**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1281
Keysight, EX-1045
Page 1281

# CHAPTER 3
# Adding SonicWALL Appliances

This chapter describes how to add SonicWALL appliances to SonicWALL ViewPoint. . This chapter contains the following sections:


- "Adding SonicWALL Appliances to SonicWALL ViewPoint" on page 37
- "Deleting SonicWALL Appliances from ViewPoint" on page 39


## Adding SonicWALL Appliances to SonicWALL ViewPoint

SonicWALL ViewPoint checks with the SonicWALL licensing server when you add an appliance, so it is important that ViewPoint has Internet access to the server.

SonicWALL ViewPoint can communicate with SonicWALL appliances through HTTP or HTTPS.  See the following sections:


- "Adding SonicWALL Appliances" on page 38
- "Modifying SonicWALL Appliance Settings" on page 39



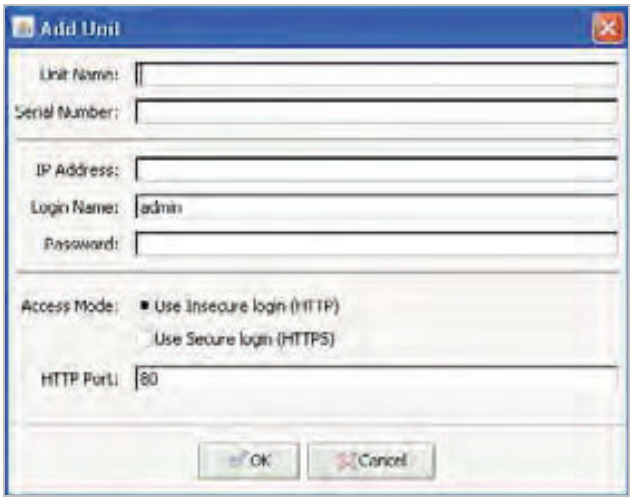


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1282
Keysight, EX-1045
Page 1282

# Adding SonicWALL Appliances

To add a SonicWALL appliance using the SonicWALL ViewPoint management interface, follow these steps:

1. Click the appliance tab that corresponds to the type of appliance that you want to add: **UTM** or **SSL-VPN**.
2. Right-click in the left pane (TreeControl pane) of the SonicWALL ViewPoint management interface and select **Add Unit**. The Add Unit dialog box appears.




3. Enter a descriptive name for the SonicWALL appliance in the **Unit Name** field.

**Note** Do not enter the single quote character (‘) in the **Unit Name** field.

4. Enter the serial number of the SonicWALL appliance in the **Serial Number** field.
5. Enter the IP address of the SonicWALL appliance in the **IP Address** field.
6. Enter the administrator login name for the SonicWALL appliance in the **Login Name** field.
7. Enter the password used to access the SonicWALL appliance in the **Password** field.
8. For **Access Mode**, select from the following:
   - If the SonicWALL appliance will be connected over HTTP, select **Use Insecure login (HTTP)**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1283
Keysight, EX-1045
Page 1283

- If the SonicWALL appliance will be connected over HTTPS, select **Use Secure login (HTTPS)**.

9. Enter the port used to connect to the SonicWALL appliance in the **HTTP(S) Port** field (default ports are HTTP: 80; HTTPS: 443).
10. Click **OK**. The new SonicWALL appliance appears in the SonicWALL ViewPoint management interface. It will have a yellow icon that indicates it has not yet been successfully acquired.

    SonicWALL ViewPoint will then attempt to set up an HTTP or HTTPS connection to access the appliance. ViewPoint then reads the appliance configuration and acquires the SonicWALL appliance for reporting. This will take a few minutes.

    After the SonicWALL appliance is successfully acquired, its icon turns blue, its configuration settings are displayed at the unit level, and its settings are saved to the database.

## Modifying SonicWALL Appliance Settings

If you make a mistake or need to change the settings of an added SonicWALL appliance, you can manually modify its settings or how it is managed.

To modify a SonicWALL appliance, perform the following steps:

1. Right-click the appliance name in the left pane of the SonicWALL ViewPoint UI and select **Modify Unit** from the pop-up menu. The Modify Unit dialog box appears.
2. The Modify Unit dialog box contains the same options as the Add Unit dialog box. For descriptions of the fields, see Adding SonicWALL Appliances to SonicWALL ViewPoint, page 37.
3. When you have finished modifying options, click **OK**. The SonicWALL appliance settings are modified.

# Deleting SonicWALL Appliances from ViewPoint

To delete a SonicWALL appliance from ViewPoint, perform the following steps:

1. Right-click on a SonicWALL appliance in the left pane and select **Delete** from the pop-up menu.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1284
Keysight, EX-1045
Page 1284

2. In the warning message that displays, click **Yes**. The SonicWALL appliance is deleted from ViewPoint.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1285
Keysight, EX-1045
Page 1285

# CHAPTER 4

# Using the SonicToday Panel

This chapter introduces the SonicToday panel in the SonicWALL ViewPoint management interface. .

This section includes the following subsections:


- "Overview of the SonicToday Panel" section on page 42
- "Editing a Component Window" section on page 42
- "Adding a Component Window" section on page 44
- "Adding More Pages" section on page 47
- "Editing and Deleting Pages" section on page 48
- "Other Features" section on page 49

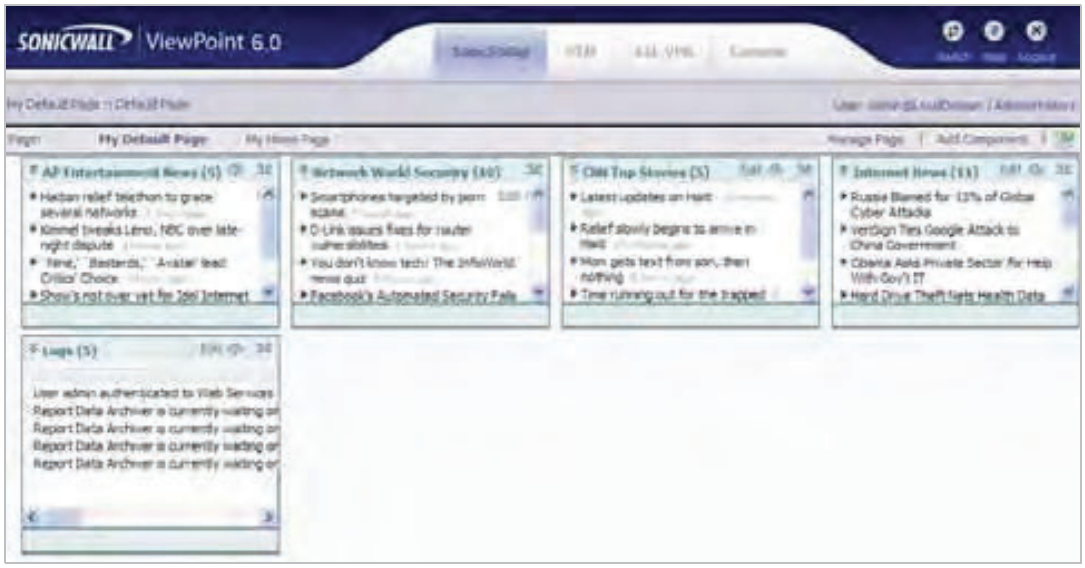



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1286
Keysight, EX-1045
Page 1286

# Overview of the SonicToday Panel

Using RSS and AJAX technology, SonicToday is a tab intended to work as a customizable dashboard where you are able to monitor the latest happenings with your SonicWALL ViewPoint 6.0 deployment, your network, the IT and Security World, as well as the rest of the world.

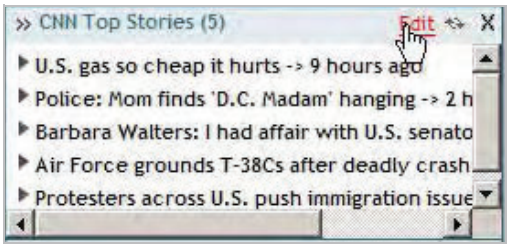
Upon initial login, you see a default **SonicToday** tab. You are able to further customize this page by configuring and adding preferred components.




# Editing a Component Window

One customizable feature of SonicToday is the ability to edit the title of any given component window. To do this:

1. Click the **Edit** link, located on the right side of the component window you wish to modify. In this example, we will modify the title of the component window “CNN Top Stories.”

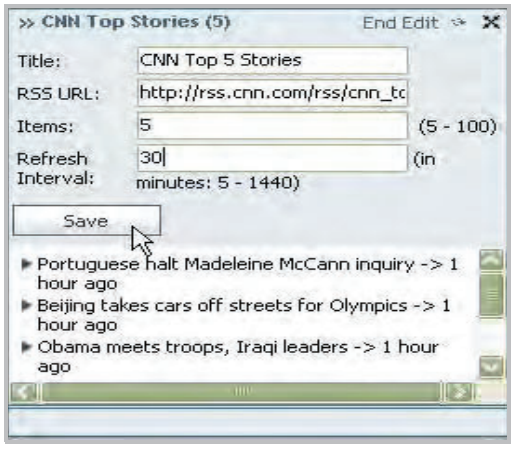





CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1287
Keysight, EX-1045
Page 1287

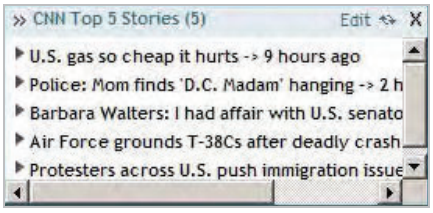
2. The component window will expand, revealing the following entries you can modify:

**Title** – The title of the component window.
**RSS URL –** The URL of the RSS Feed the current component window updates from.
**Items –** The number of items to be displayed on the component window.
**Refresh Interval** – The frequency of time the component window will refresh the RSS Feed.




In this example, we will change the title to “CNN Top 5 Stories.” For **Items**, we specify that we want five items shown in the component window, and we want the **Refresh Interval** to occur every 30 minutes. Click **Save** to save your changes and exit the component window.

The changes will update the component window immediately.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1288
Keysight, EX-1045
Page 1288

# Adding a Component Window

Another way to fully customize your SonicToday dashboard is by adding a component window specifically to your preferences.

Note that no component containing the same content can be added more than once in the SonicToday dashboard.

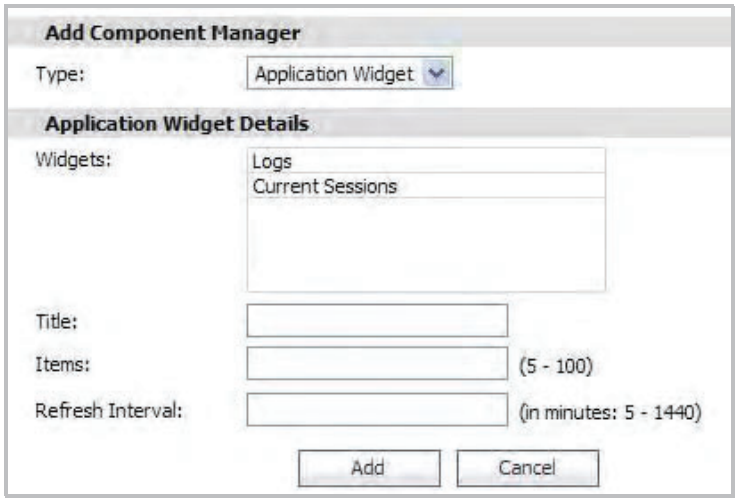
In this section, there are different component windows you can add:

- “Application Widget” section on page 44
- “RSS Feed” section on page 46

## Application Widget

The application widget specifically details Logs and Current Sessions in SonicWALL ViewPoint 6.0. The convenience of this new widget is that it enables you to keep track of all these different details from the SonicToday dashboard page, rather than navigating through other tabs. To add the application widget:

1. Click **Add Component** to bring up the Add Component Manager dialogue box. Select **Application Widget** from the ‘Type’ drop-down list.

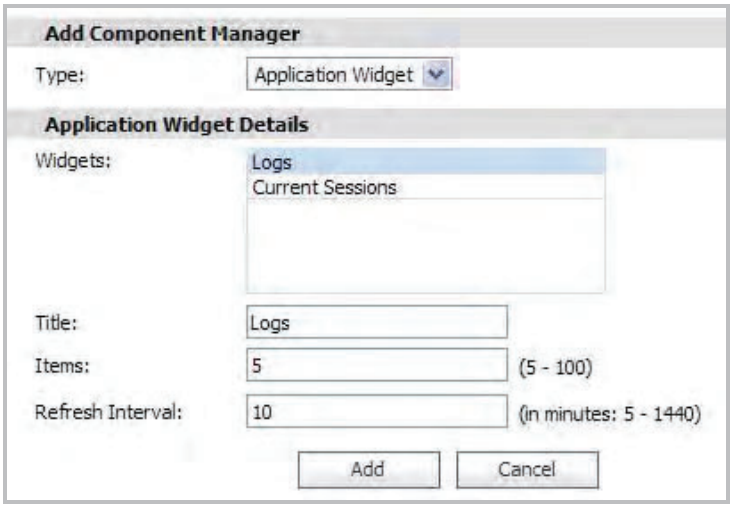




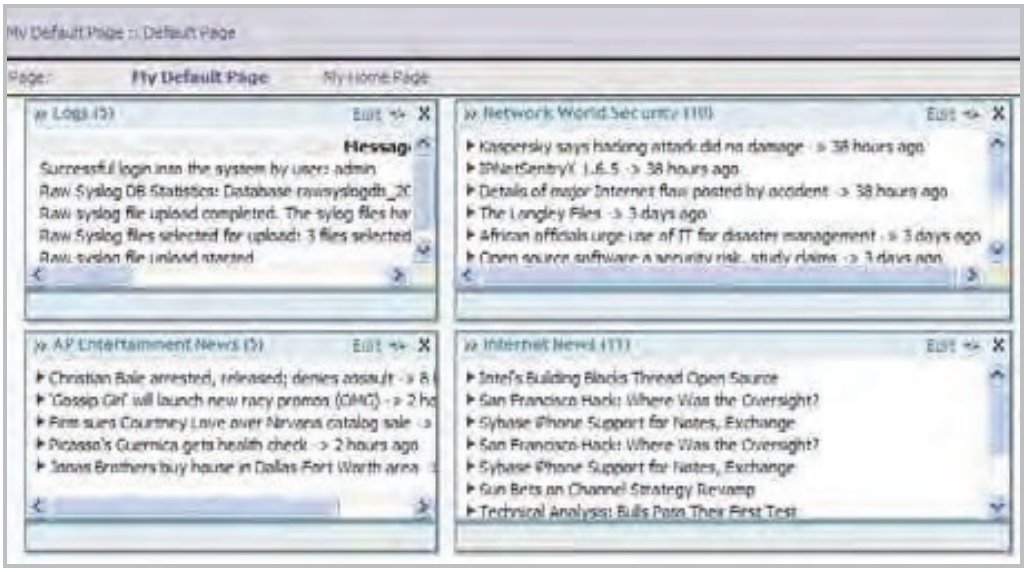

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1289
Keysight, EX-1045
Page 1289

2. Specify what type of **Widget** you want in the component. The **Title** will default to the Widget you choose, but you may customize this if you prefer. You also will indicate how many **Items** you want to be shown on the component window, as well as the **Refresh Interval**.

   In this example, we will add a widget that monitors Logs, displaying the latest five everyten minutes.




3. Click **Add** when finished specifying entries. The component window is added to the SonicToday dashboard.





CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1290
Keysight, EX-1045
Page 1290

# RSS Feed

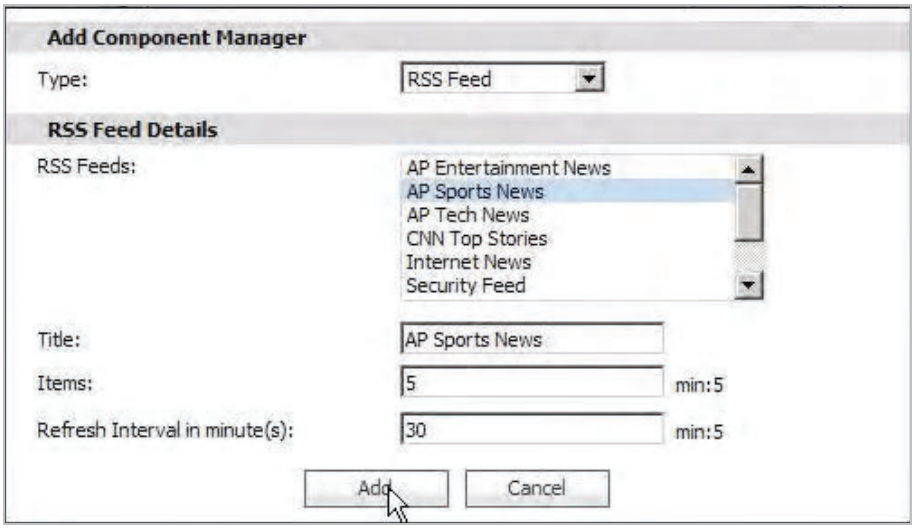
RSS Feed is a component window designed to keep you updated with what is going on in the IT and Security World, as well as all around the globe. This section contains procedures for customizing an RSS Feed component window on your SonicToday dashboard.

**To choose a Predefined RSS Feed**:

1. Click **Add Component** to bring up the Add Component Manager dialogue box.
2. Select **RSS Feed** from the 'Type' drop-down list. This will automatically bring up a list of predefined RSS Feeds you may choose from.

   The **Title** will default to the Alert Type you choose, but you may customize this if you prefer. You also will indicate how many **Items** you want to be shown on the component window, as well as the **Refresh Interval**.

   In this example, we will select 'AP Sports News,' displaying the first five items every 30 minutes on the component window.




3. Click **Add** when you are finished. This will add the new RSS Feed component window to your SonicToday dashboard.

**To Choose a Custom RSS Feed**:

1. Click **Add Component** to bring up the Add Component Manager dialogue box.
2. Select **RSS Feed** from the 'Type' drop-down list. This will automatically bring up a list of predefined RSS Feeds you may choose from.
3. Scroll to the bottom of the predefined list and select **Custom RSS Feed...** Enter the URL of the RSS Feed you would like on your component window.

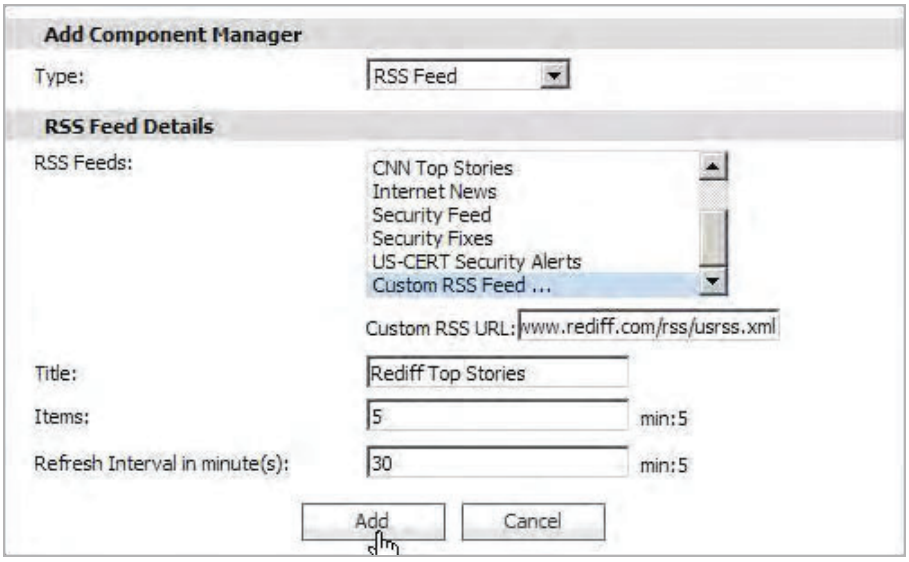


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1291
Keysight, EX-1045
Page 1291

**Note** To search a large directory of available RSS Feeds, navigate to: http://www.rsfeeds.com/

4. Enter the **Title** for this custom RSS Feed page. Also indicate how many **Items** you want to be shown on the component window, as well as the **Refresh Interval**.

   In this example, we will choose 'Rediff Top Stories,' displaying the first five items every 30 minutes on the component window.




5. Click **Add** when you are finished. This will add the new RSS Feed component window to your SonicToday dashboard.

# Adding More Pages

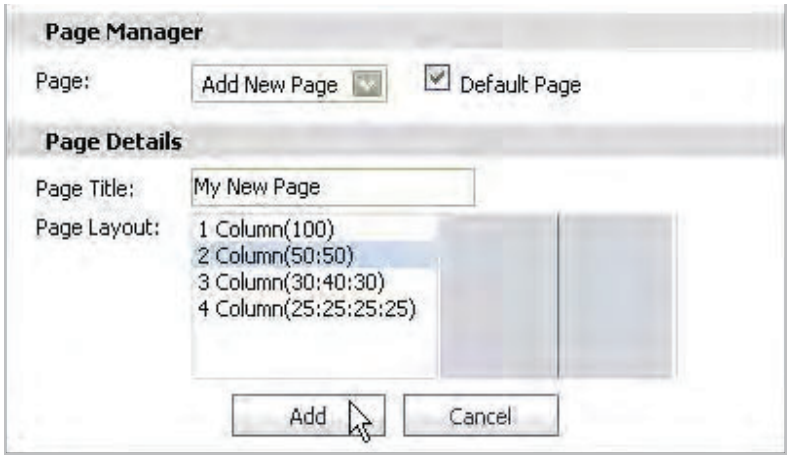
SonicToday allows you to create more pages in addition to your default dashboard page. Note that only one page may be designated as your SonicToday default page. As soon as a new page is marked as the default, any previous default page settings are overwritten. To create a new page:

1. Click **Manage Page** from the toolbar to bring up the Page Manager.
2. In the 'Page' section, select **Add New Page** from the drop-down list.
3. Name your new page under 'Page Title.'
4. Select the layout of your page under 'Page Layout.' A thumbnail image pops up alongside each option to assist you.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1292
Keysight, EX-1045
Page 1292

5. You also have the option of making this your default page, simply by placing a checkmark in the box labeled ‘Default Page.’




6. Click **Add** when you are finished. The toolbar now displays the newly added page.
In this example, we titled the new page ‘News.’




You can now add and customize component windows to navigate between pages.

# Editing and Deleting Pages

To edit a page, click **Manage Page** from the toolbar. Select the page you wish to edit, make your changes, and click **Edit** to finish.

To delete a page, click **Manage Page** from the toolbar. Select the page you wish to delete and click **Delete**. Click **OK** to finish.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1293
Keysight, EX-1045
Page 1293

# Other Features

See the following sections:


- AutoHide, page 49
- Page Selector, page 49
- Component Height Resize, page 50
- Manual Refresh, page 50
- Removing or Deleting a Component, page 50
- Minimizing or Maximizing a Component, page 50


## AutoHide

AutoHide is a feature you customize by turning on or off. When AutoHide is turned on, the control bar will hide after an interval of two seconds when the mouse is moved away from the control bar. When AutoHide is turned off, the control bar always appears on the SonicToday dashboard.

To turn AutoHide on, click the **Off** icon  .

To turn AutoHide off, click the **On** icon 

## Page Selector

Whenever the number of pages added to the SonicToday dashboard exceeds five, a page selector bar appears at the top of the main window with left and right arrows. The arrows can be used to scroll across different pages in both directions. By default, the selector is scrolled to a point where the default page appears on it. Any page can be selected by clicking on the page title.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1294
Keysight, EX-1045
Page 1294

## Component Height Resize

The height of a component can be increased and decreased by stretching or shrinking the resize cursor on the status bar when the mouse is moved over the status bar.




## Manual Refresh

Aside from the automatic refresh, which you configure in the "Editing a Component Window" section on page 42, you can force a refresh on the component window by clicking the **refresh** icon on the component window header.

## Removing or Deleting a Component

Any component window can be removed or deleted from the page by clicking the **close** icon on the component window header.

## Minimizing or Maximizing a Component

Each component can be in minimized or maximized state. The components are loaded in the page with the state they were saved in the database.

To minimize a component window, click the **minimize** icon in the component window header.

To maximize a component window, click the **maximize** icon in the component window header.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1295
Keysight, EX-1045
Page 1295

# CHAPTER 5
# Configuring User Settings

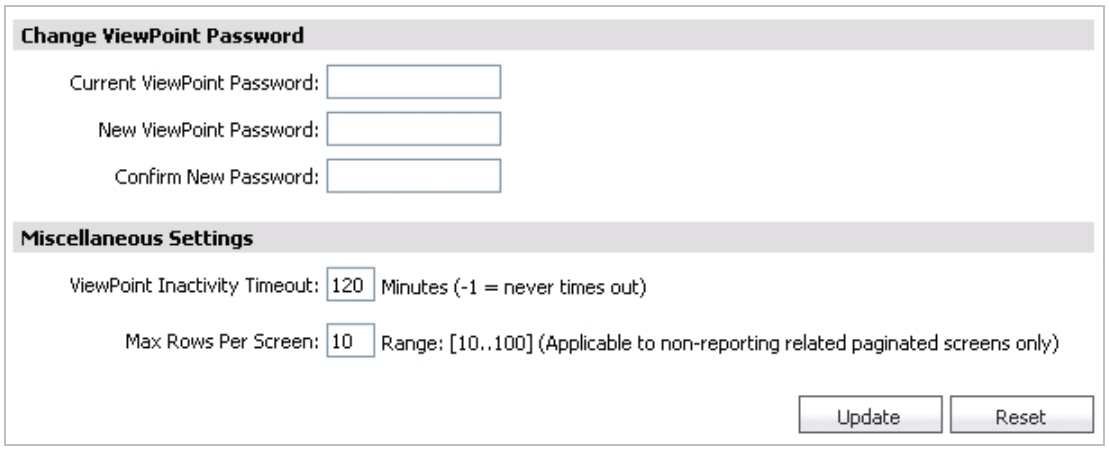
This chapter describes how to configure the user settings that are available in the Console panel on the User Settings screens.

This chapter includes the following sections:


- "Configuring General Settings" section on page 51
- "Configuring Reports Settings" section on page 53


## Configuring General Settings

This section describes the **User Settings > General** page, which provides a way to change the ViewPoint administrator password, the ViewPoint inactivity Timeout, and pagination settings.







CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1296
Keysight, EX-1045
Page 1296

Perform the following steps:

1. Enter the existing SonicWALL ViewPoint password in the **Current ViewPoint Password** field.
2. Enter the new SonicWALL ViewPoint password in the **New ViewPoint Password** field.
3. Reenter the new password in the **Confirm New Password** field.



**Note** Password fields will be grayed out for users on a Remote Domain.

4. The ViewPoint Inactivity Timeout period specifies how long SonicWALL ViewPoint waits before logging out an inactive user. To prevent someone from accessing the SonicWALL ViewPoint UI when SonicWALL ViewPoint users are away from their desks, enter an appropriate value in the **ViewPoint Inactivity Timeout** field. You can disable automatic logout completely by entering a “-1” in this field. The minimum is 5 minutes and the maximum is 120 minutes.
5. Select a value between 10 and 100 in the **Max Rows Per Screen** field. This value applies only to non-reporting related paginated screens.
6. When you are finished, click **Update**. The settings are changed. To clear all screen settings and start over, click **Reset**.



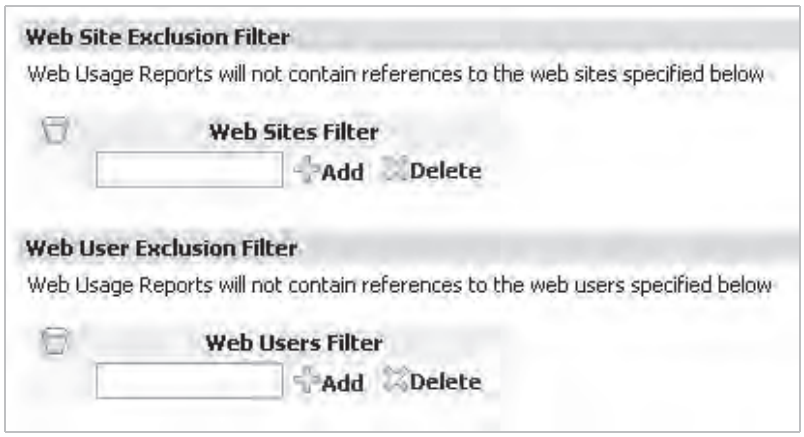
**Note** The maximum size of the SonicWALL ViewPoint User ID is 24 alphanumeric characters. The password is one-way hashed and any password of any length can be hashed into a fixed 32 character long internal password.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1297
Keysight, EX-1045
Page 1297

# Configuring Reports Settings

The User Settings > Reports page on the Console panel provides settings for the Web Site Exclusion Filter and Web User Exclusion Filter. Web Usage reports will not contain references to the Web sites or users specified on this page.




The following Web Usage reports are affected by the Web Site and Web User Exclusion Filters:

- Web Usage > Summary
- Web Usage > Top Sites
- Web Usage > Top Users
- Web Usage > By User
- Web Usage > By Site
- Web Usage > By Category
- Web Usage > Over Time
- Web Usage > Top Sites Over Time
- Web Usage > Top Users Over Time
- Web Usage > By User Over Time
- Web Usage > By Category Over Time



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1298
Keysight, EX-1045
Page 1298

# Adding Web Sites to the Filter List

When entering the Web site to exclude, type only the site name. The filter will search for the exact value provided. In the reports, only the site name is listed, without the http:// or www prefix. So for example, http://site1.sonicwall.com would not find a match in any reports because it would be listed in the reports simply as site1.sonicwall.com.

To add a Web site to the Web Sites Filter list, perform the following steps:

1. On the Console > User Settings > Reports page, type the Web site to be excluded into the **Web Sites Filter** field.

   Enter the Web site without the http:// or www prefix.

2. Click the **Add** button.

# Deleting Web Sites from the Filter List

To remove a Web site from the Web Sites Filter list, perform the following steps:

1. On the Console > User Settings > Reports page, select the checkbox next to the Web site to be removed from the exclusion list. To select all sites in the list, select the **Select All** checkbox.
2. Click the **Delete** button.

# Adding Web Users to the Filter List

To add a user to the Web Users Filter list, perform the following steps:

1. On the Console > User Settings > Reports page, type the user name to be excluded into the **Web Users Filter** field.

   Enter the user name without the domain.

2. Click the **Add** button.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1299
Keysight, EX-1045
Page 1299

# Deleting Web Sites from the Filter List

To remove a Web user from the Web Users Filter list, perform the following steps:

1. On the Console > User Settings > Reports page, select the checkbox next to the user to be removed from the exclusion list. To select all users in the list, select the **Select All** checkbox.
2. Click the **Delete** button.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1300
Keysight, EX-1045
Page 1300

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1301
Keysight, EX-1045
Page 1301

# CHAPTER 6
# Configuring Log Settings

This section describes how to configure Log Settings. This includes adjusting settings on deleting log messages after a certain period of time, and setting criteria for viewing logs.

This chapter includes the following sections:

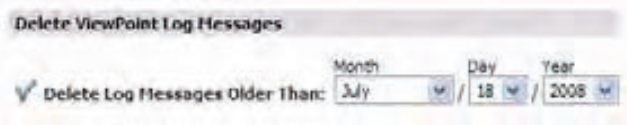

- "Configuration" section on page 57
- "View Log" section on page 58


## Configuration

The Log > Configuration screen provides a way to delete log messages older than a specific date.

To delete ViewPoint log messages, perform the following steps:

1. Click the **Console** tab, expand the **Log** tree, and click **Configuration**. The Configuration page displays.




2. Select the month, day, and year from the drop down menu.
3. Click **Delete Log Messages Older Than**.

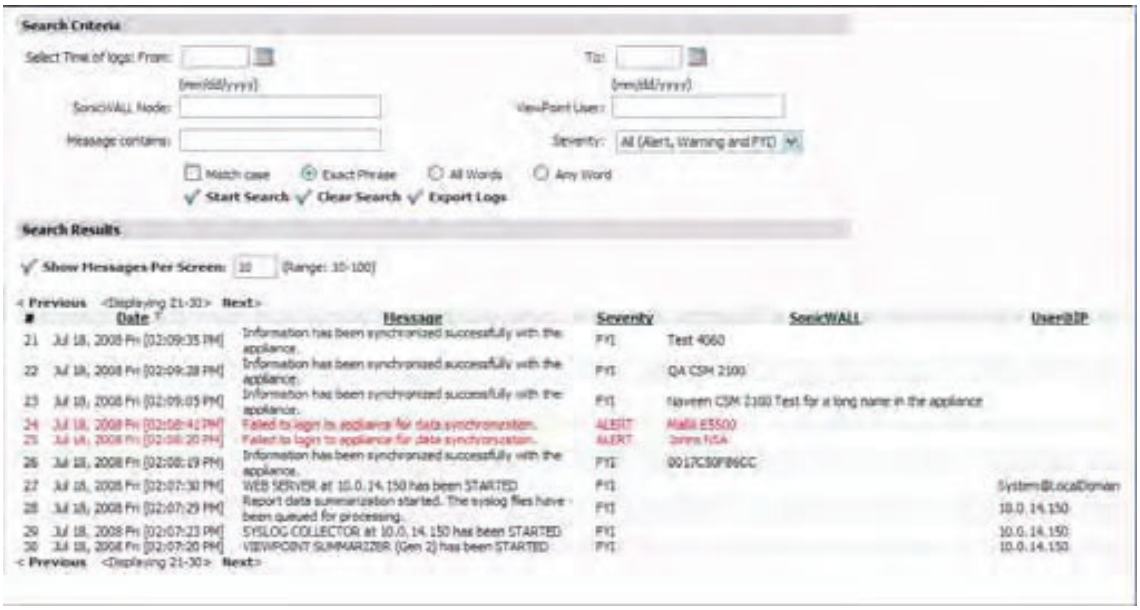



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1302
Keysight, EX-1045
Page 1302

# View Log

The SonicWALL ViewPoint log keeps track of changes made within the SonicWALL ViewPoint UI, logins, failed logins, logouts, password changes, scheduled tasks, failed tasks, completed tasks, raw syslog database size, syslog message uploads, and time spent summarizing syslog data. To view the SonicWALL ViewPoint log, perform the following steps:

1. Click the Console tab, expand the Log tree, and click **View Log**. The View Log page displays.




2. Each log entry contains the following fields:
   - **#**—specifies the number of the log entry.
   - **Date**—specifies the date of the log entry.
   - **Message**—contains a description of the event.
   - **Severity**—displays the severity of the event (Alert, Warning, or FYI).
   - **SonicWALL**—specifies the name of the SonicWALL appliance that generated the event (if applicable).
   - **User@IP**—specifies the user name and IP address.
3. To narrow the search, configure some of the following criteria:

**Tip** You can press Enter to navigate from one form element to the next in this section.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1303
Keysight, EX-1045
Page 1303

- **Select Time of logs**—displays all log entries for a specified range of dates.
- **SonicWALL Node**—displays all log entries associated with the specified SonicWALL appliance.
- **ViewPoint User**—displays all log entries with the specified user.
- **Message contains**—displays all log entries that contain the specified text. This input field provides an auto-suggest functionality that uses existing log message text to predict what you want to type. It fills in the field with the suggested text and you can either press **Tab** to accept it or keep typing. Different suggestions will appear as you continue to type if log messages match your input.
- **Severity**—displays log entries with the matching severity level:
  - All (Alert, Warning, and FYI)–where FYI mean “For Your Information”
  - Alert and Warning
  - Alert
- Select the **Match case** checkbox to make the **SonicWALL Node**, **ViewPoint User**, and **Message contains** search fields case sensitive.
- Select one of **Exact Phrase**, **All Words**, or **Any Word**.
  - **Exact Phrase** matches a log entry that contains exactly what you typed in the **Message contains** field
  - **All Words** matches a log entry that contains all the words you typed in the **Message contains** field, but the words can be non-consecutive or in any order
  - **Any Word** matches a log entry that contains any of the words you typed in the **Message contains** field

4. To view the results of your search criteria, click **Start Search**. To clear all values from the input fields and start over, click **Clear Search**. To save the results as an HTML file on your system, click **Export Logs** and follow the on-screen instructions.
5. To configure how many messages are shown per screen, enter a new value between 10 and 100 in the **Show Messages Per Screen** field. (default: 10). Click **Next** to display the next page, or click **Previous** to display the preceding page.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1304
Keysight, EX-1045
Page 1304

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1305
Keysight, EX-1045
Page 1305

# CHAPTER 7
# Configuring Management Settings

This chapter describes the settings available on the Console panel in the Management section. The following sections are found in this chapter:


- "Settings" section on page 61
- "Alert Settings" section on page 64
- "Sessions" section on page 65
- "Database Maintenance" section on page 66


## Settings

On the Console > Management >Settings page, you can  configure email settings, set the system debug level, synchronize model codes information, and configure password security settings..

This section describes the following Settings topics:


- "Configuring Email Settings" on page 62
- "Configuring System Debug Level" on page 62
- "Enforcing Password Security" on page 63
- "Synchronizing Model Codes" on page 63





CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1306
Keysight, EX-1045
Page 1306

# Configuring Email Settings

An SMTP server and an email address are required for sending ViewPoint reports.

If the Mail Server settings are not configured correctly, you will not receive important email notifications, such as:

- System alerts for your SonicWALL ViewPoint deployment performance
- Availability of product updates, hot fixes, or patches
- Scheduled Reports

To configure these email settings:

1. Click the **Console** tab.
2. Expand the **Management** tree and click **Settings**. The Settings page displays.
3. Type the IP address of the Simple Mail Transfer Protocol (SMTP) server into the **SMTP Server** field. This server can be the same one that is normally used for email in your network.
4. Type the email account name and domain that will appear in messages sent from the SonicWALL ViewPoint into the **ViewPoint Sender's e-Mail Address** field.
5. When finished in the Settings page, click **Update**. To clear the screen settings and start over, click **Reset**.

# Configuring System Debug Level

SonicWALL ViewPoint provides the **System Debug level** option to control the debug messages sent to the log file.

To configure this setting:

1. Select a debug level from the **System Debug level** drop-down list. The range is 0-3 where a level of 0 provides no debug log messages and a level of 3 provides the maximum number of debug messages.
2. When finished in the Settings page, click **Update**. To clear the screen settings and start over, click **Reset**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1307
Keysight, EX-1045
Page 1307

# Enforcing Password Security

SonicWALL ViewPoint supports enforced password rotation for enhanced security compliance.

To enable and configure enforced password rotation:

1. Select the **Enforce Password Security** checkbox.
2. In the **Number of days to force password change** field, enter a value. The default is 90. SonicWALL ViewPoint will prompt the administrator to change the admin account password after the specified number of days.
3. When finished in the Settings page, click **Update**. To clear the screen settings and start over, click **Reset**.

# Synchronizing Model Codes

The Sync Model Codes feature accommodates new SonicWALL product introductions without the need for ViewPoint update. When SonicWALL updates the the corporate server (MySonicWALL) with a new product code, it then becomes available to ViewPoint. The task is scheduled to run every 24 hours and is also available manually.

To synchronize model codes immediately:

1. On the Console > Management > Settings page, click **Sync Model Codes information now**.
2. A short time later the page is updated to display the synchronization status at the top.



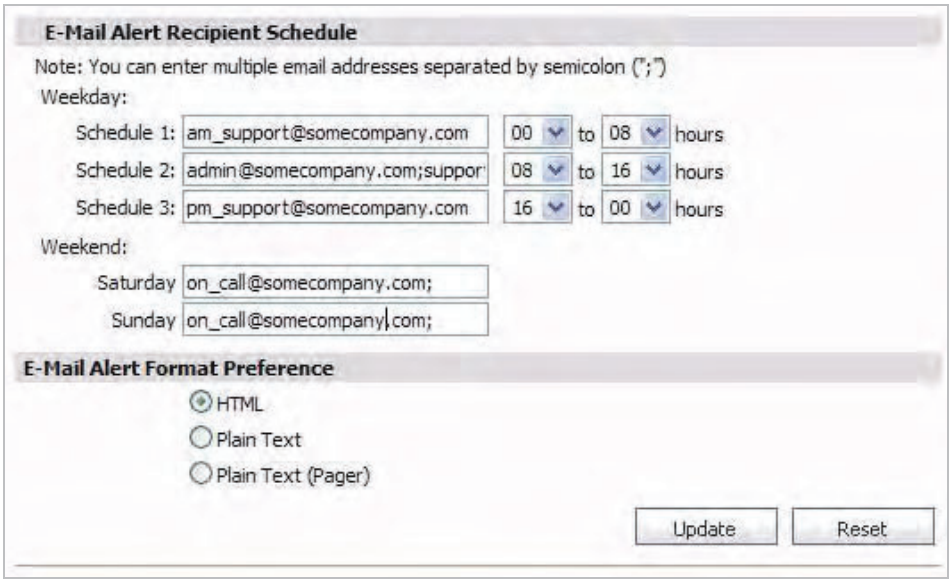
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1308
Keysight, EX-1045
Page 1308

# Alert Settings

The Alert Settings page specifies which email addresses receive email alerts and notifications during specific times.

To configure the alert notification settings, perform the following steps:

1. Click the **Console** tab, expand the **Management** tree and click **Alert Settings**. The Alert Settings page displays.




2. Configure the email address(es) that will receive notifications and the times that they will receive them:
   - **Schedule 1**—Specifies who will receive notifications during the first weekday schedule. Enter one or more email addresses (separated by commas) and specify the start and end time for the shift.
   - **Schedule 2**—Specifies who will receive notifications during the second weekday schedule. Enter one or more email addresses (separated by commas) and specify the start and end time for the shift.
   - **Schedule 3**—Specifies who will receive notifications during the third weekday schedule. Enter one or more email addresses (separated by commas) and specify the start and end time for the shift.
   - **Saturday**—Specifies who will receive notifications on Saturday. Enter one or more email addresses (separated by commas) and specify the start and end time for the shift.
   - **Sunday**—Specifies who will receive notifications on Sunday. Enter one or more email addresses (separated by commas) and specify the start and end time for the shift.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1309
Keysight, EX-1045
Page 1309

3. Select whether the email alert will be sent as **HTML**, **Plain Text**, or **Plain Text (Pager**). The Pager setting sends a very short email to ensure that the email is not cut off by the character limits of some pagers.
4. When you are finished, click **Update**. The settings are saved.

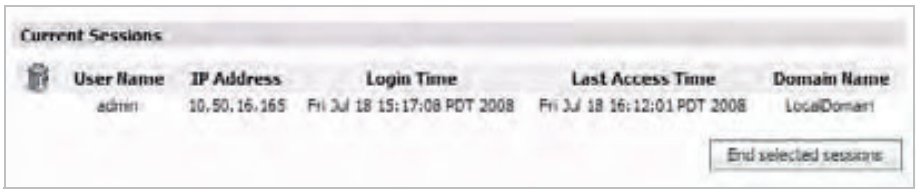
# Sessions

The Sessions page of the Management section of the ViewPoint Console allows you to view session statistics for currently logged in ViewPoint users and to end selected sessions.

## Managing Sessions

On occasion, it may be necessary to log off other user sessions. To do this, perform the following steps:

1. Click the **Console** tab, expand the **Management** tree and click **Sessions**. The Sessions page displays.




2. When more than one session is active, a checkbox is displayed next to each row. Select the check box of each user to log off and click **End selected sessions**.

   The selected users are logged off.

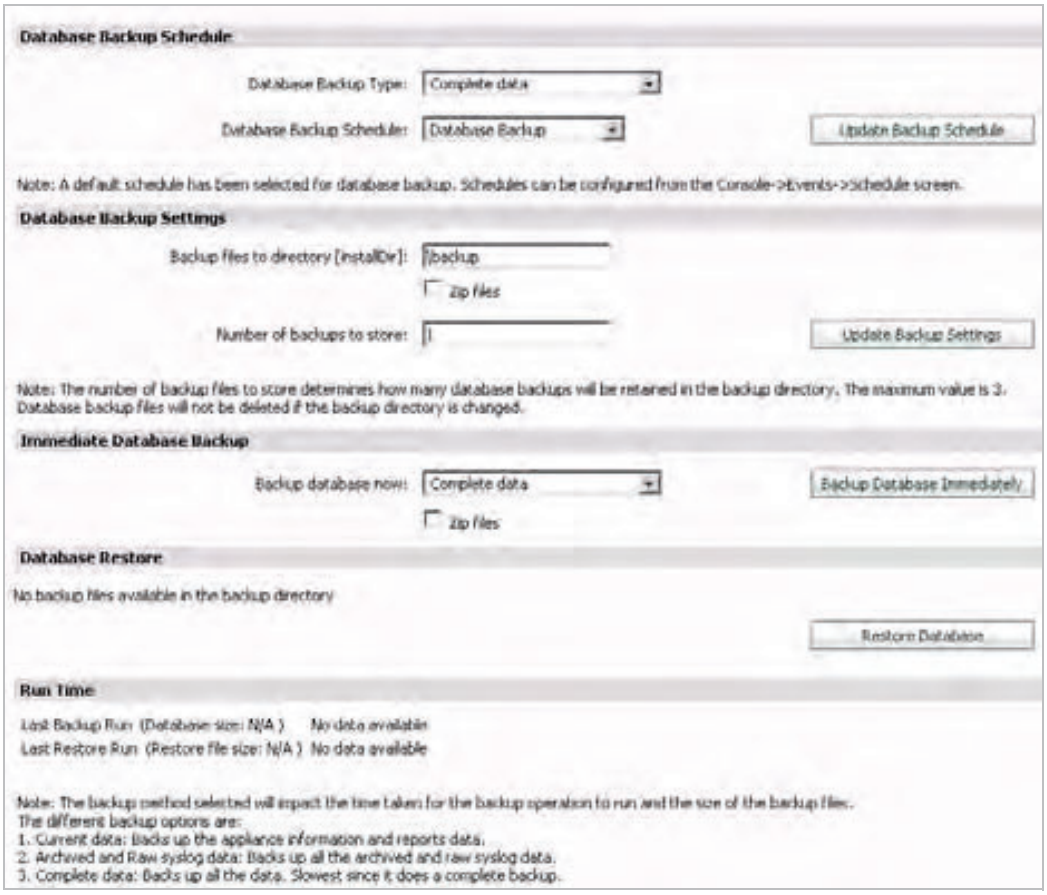


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1310
Keysight, EX-1045
Page 1310

# Database Maintenance

The Database Maintenance page allows you to back up the MySQL databases used by SonicWALL ViewPoint. This screen is not applicable to deployments using SQL Server.

**Note** The Console > Management > Database Maintenance page only appears in the management interface when a MySQL database is being used.




You can configure the type of backup, schedule for periodic backups, folder for backup storage, and number of backups (up to 3) to keep. You can also perform an immediate database backup from this page. Existing backups of the database are listed, and you can select from them to restore your databases.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1311
Keysight, EX-1045
Page 1311

See the following sections:


- Configuring Backup Schedule and Settings, page 67
- Backing Up a Database Immediately, page 68
- Restoring a Database Backup, page 68


# Configuring Backup Schedule and Settings

To configure the database backup schedule and settings, perform the following steps:

1. Click the **Console** tab, expand the **Management** tree, and click **Database Maintenance**. The Database Maintenance page displays.
2. Under Database Backup Schedule, select one of the following from the **Database Backup Type** drop-down list:
   - **Current data** – Backs up system information and all data in **sgmsdb** for the current month; **sgmsdb** contains summarized report data
   - **Archived and Raw syslog data** – Backs up the archived data that is moved from **sgmsdb** to other files at the end of every month, and backs up raw syslog data
   - **Complete data** – Backs up all data including **sgmsdb** and all archived data and raw syslog data; this option requires the most time
3. Select the desired backup schedule from the **Database Backup Schedule** drop-down list. You can select a pre-configured schedule or a custom schedule, which you can configure in the Console > Events > Schedule screen.
4. When finished selecting options under Database Backup Schedule, click the **Update Backup Schedule** button.
5. Under Database Backup Settings in the **Backup files to directory [installDir]** field, enter the folder name in which you want to store the backup files.
6. Select the **Zip files** checkbox if you want the backup to be compressed and stored as a **.zip** file.
7. In the **Number of backups to store** field, enter the number of backups you want to store. The maximum is 3. When the maximum number of backups is reached in the configured folder, the oldest one will be removed when a new backup is created. If the folder is changed, existing backups in the previous folder will not be deleted.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1312
Keysight, EX-1045
Page 1312

8. When finished selecting options under Database Backup Settings, Select the **Zip files** checkbox if you want the backup to be compressed and stored as a **.zip** file.
9. When finished selecting options under Database Backup Settings, click the **Update Backup Settings** button.

# Backing Up a Database Immediately

To perform an interactive backup of a database, complete the following steps:

1. On the Console > Management > Database Maintenance page, under **Immediate Database Backup**, select the type of backup from the **Backup database now** drop-down list. You can select one of the following types:
   - **Current data** – Backs up system information and all data in **sgmsdb** for the current month; **sgmsdb** contains summarized report data
   - **Archived and Raw syslog data** – Backs up the archived data that is moved from **sgmsdb** to other files at the end of every month, and backs up raw syslog data
   - **Complete data** – Backs up all data including **sgmsdb** and all archived data and raw syslog data; this option requires the most time
2. Select the **Zip files** checkbox if you want the backup to be compressed and stored as a **.zip** file.
3. Click the **Backup Database Immediately** button.
4. In the confirmation dialog box, click **OK**.

# Restoring a Database Backup

This feature allows the administrator to restore a previously backed-up database file.



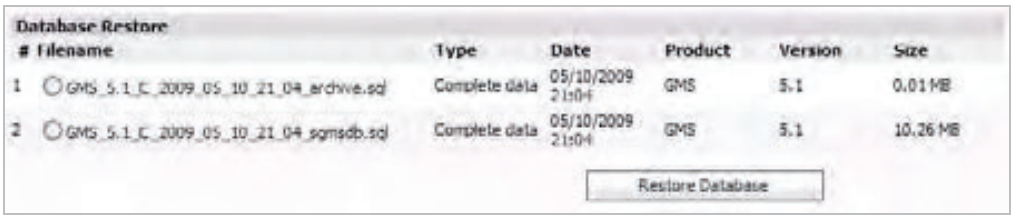
**Note** All services except the Web Server and the Database Service should be manually stopped before restoration is started to avoid corruption of data.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1313
Keysight, EX-1045
Page 1313

To restore your database with one of your backups, perform the following steps:

1. On the **Console > Management > Database Maintenance** page, under **Database Restore**, select the radio button for the backup that you want to restore.




2. Click the **Restore Database** button.
3. In the confirmation dialog box, click **OK**.
4. You must restart the Web Server service manually after the backup is completed.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1314
Keysight, EX-1045
Page 1314

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1315
Keysight, EX-1045
Page 1315

# CHAPTER 8
# Managing Reports in the Console Panel

This section describes how to configure reporting settings on the Console panel. These include how often the summary information is updated, the number of days that summary information is stored, and the number of days that raw data is stored.

The following sections are included in this chapter:


- "Settings" section on page 71
- "Summarizer" section on page 73
- "Email/Archive" section on page 81
- "Scheduled Reports" section on page 82
- "Management" section on page 87


## Settings

The Settings page under Reports on the Console panel provides a check box for enabling the sort option in report tables. You can also specify the number of appliances which can have Log Viewer enabled at the same time.

See the following:


- "Enabling Report Table Sorting" section on page 72
- "Controlling the Number of Appliances with Log Viewer Enabled" section on page 72





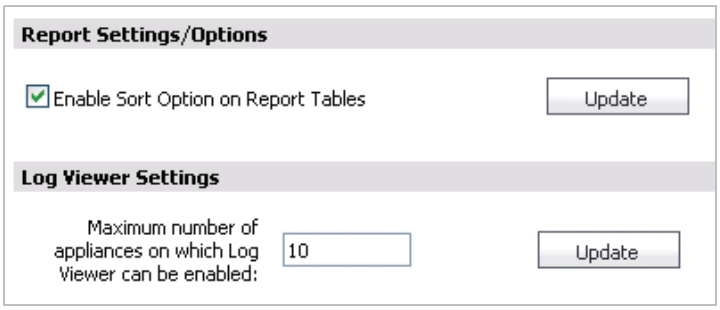
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1316
Keysight, EX-1045
Page 1316

# Enabling Report Table Sorting

The Report Settings/Options section of the Console > Reports > Settings page provides a checkbox to enable the sort option on report tables.

To enable or disable the sort option for report tables, perform the following steps:

1. Click the **Console** tab, expand the **Reports** tree and click **Settings**.




2. To enable the report table sort option, select the **Enable Sort Option on Report Tables** checkbox. To disable sorting, clear the checkbox.
3. Click **Update**.

# Controlling the Number of Appliances with Log Viewer Enabled

You can control the maximum number of managed appliances for which Log Viewer can be enabled. The default setting allows Log Viewer to be enabled on up to five appliances. Because enabling Log Viewer causes raw syslog data uploading, it is resource intensive. Use care in increasing this number, and when enabling Log Viewer on systems.

Log Viewer must be enabled on an appliance in order to use Custom Reports. Custom Reports are available for UTM and SSL-VPN appliances. For more information about Custom Reports, see the following:

- “Using Custom Reports on UTM Appliances” section on page 163

To change the number of appliances for which Log Viewer can be enabled:

1. On the **Console** panel, navigate to **Reports > Settings**.
2. Under **Log Viewer Settings**, in the **Maximum number of appliances on which Log Viewer can be enabled** field, enter the number of appliances for which Log Viewer can be enabled. The default is five.
3. Click **Update**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1317
Keysight, EX-1045
Page 1317



**Note** Limiting the number of appliances for which the Log Viewer is enabled will increase the overall performance of your SonicWALL ViewPoint system.

# Summarizer

This section contains the following subsections:


- "About Summary Data in Reports" on page 73
- "Summarizer Settings and Summarization Interval" on page 73
- "Configuring the Syslog Deletion Schedule Settings" on page 78
- "Configuring Host Name Resolution" on page 79


## About Summary Data in Reports

These reports are constructed from the most current available summary data. In order to create summary data, the ViewPoint Reporting Module must parse the raw data files.

When configuring ViewPoint Reporting using the screens on the Console panel under Reports, you can select the amount of summary information to store. These settings affect the database size, be sure there is adequate disk space to accommodate the settings you choose.

Additionally, you can select the number of days that raw syslog data is stored. The raw data is made up of information for every connection. Depending on the amount of traffic, this can quickly consume an enormous amount of space in the database. ViewPoint creates a new 2 GB database for raw syslog data everyday. Be very careful when selecting how much raw information to store. For information on configuring raw data storage, see "Enabling Report Table Sorting" section on page 72.

## Summarizer Settings and Summarization Interval

SonicWALL appliances send their syslog packets to SonicWALL ViewPoint via UDP packets. When summarization is enabled, the Summarizer will process those files and store the data in the summary databases at the interval you specify.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1318
Keysight, EX-1045
Page 1318

See the following sections:

- “Enabling Report Summarization” section on page 74
- “Setting the Reports Data Summarization Interval” section on page 74
- “Using Summarize Now” section on page 76

## Enabling Report Summarization

To globally enable the summarization of report data, which is necessary for viewing reports, perform the following:

1. On the **Console** panel, navigate to **Reports > Summarizer**.
2. Under **Summarizer Settings**, select the **Enable Report Summarization** checkbox.
3. Click **Update**.

## Setting the Reports Data Summarization Interval

The Summarizer will process syslog data sent from SonicWALL appliances and store the processed data in the summary databases at the interval you specify. When an appliance is configured to communicate with ViewPoint, you need to verify that the summarizer is scheduled to collect and process data for this unit at an appropriate interval.

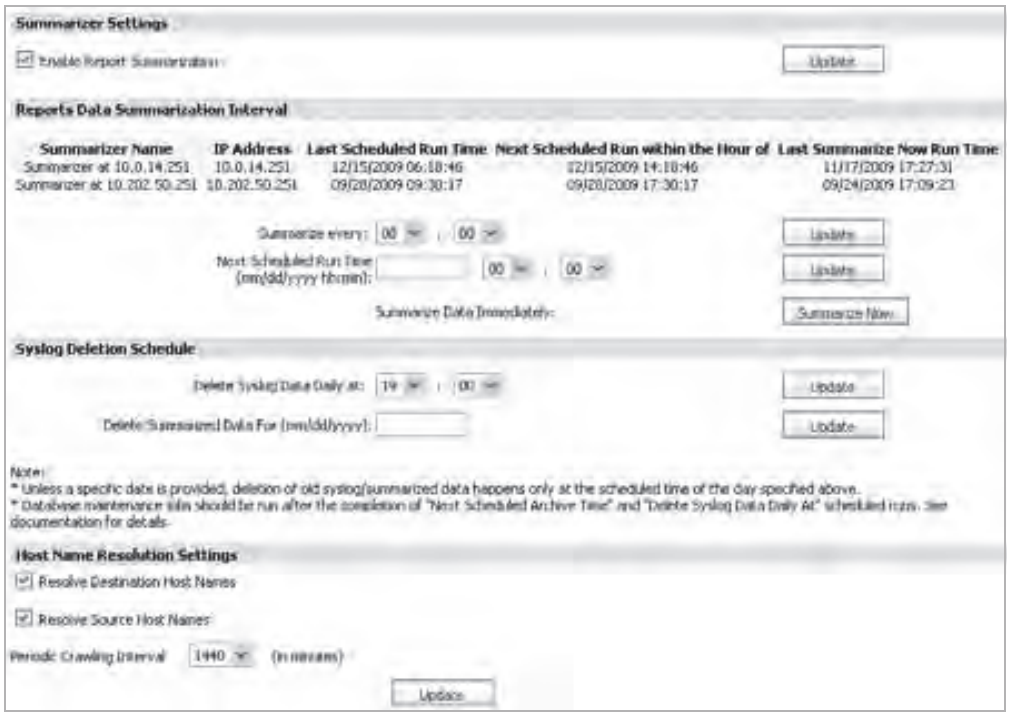
To configure reports for summarization, see the “Selecting Reports for Summarization” section on page 137 in the Scheduling and Configuring Reports chapter.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1319
Keysight, EX-1045
Page 1319

To configure the summarization interval, perform the following steps:

1. Click the **Console** tab, expand the **Reports** tree and click **Summarizer**. The Summarizer page displays.




2. Under Reports Data Summarization Interval, important information about the Summarizer is displayed. Use the **Summarize every** drop-down lists to specify how often in hours and minutes the ViewPoint Reporting Module should process syslog data and update summary information.
3. Click the **Update** button to the right of this field.
4. To specify the next summarization time, enter a date in the form mm/dd/yyyy in the **Next Scheduled Run Time** field, and select the hour and minute values from the drop-down lists.
5. Click the **Update** button to the right of this field.
6. To update the summary information now, click the **Summarize Now** button. SonicWALL ViewPoint will automatically process the latest information and make it available for immediate viewing.

**Note** This will not affect the normally scheduled summarization updates on ViewPoint.

For more information about using and verifying the Summarize Now option, see the "Using Summarize Now" section on page 76.



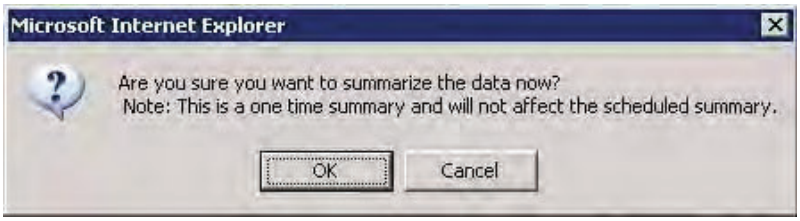
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1320
Keysight, EX-1045
Page 1320

# Using Summarize Now

The Summarize Now feature allows the administrator to create instant summary reports without affecting the regularly scheduled summary reports. You can use Summarize Now to test that the Summarizer is gathering data for a managed unit. The SonicWALL ViewPoint Summarize Now feature is located in the **Console** tab under **Reports > Summarizer**. The SonicWALL ViewPoint Summarizer creates summary reports by default every 8 hours. Summary reports can be configured by the administrator to occur every 15 minutes to every 24 hours.

To use the Summarize Now feature, perform the following tasks:

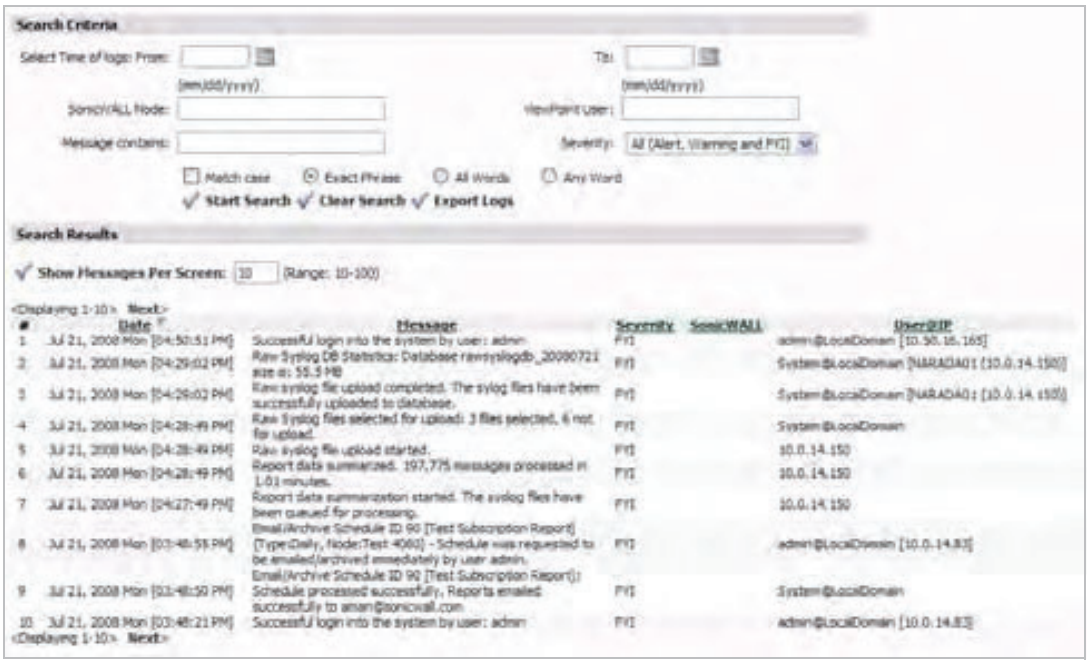
1. Click the **Console** tab, expand the **Reports** tree and click **Summarizer**. Click the **Summarize Now** button.
2. You will see a pop-up window verifying that you want to summarize the data now. Summarizing data using **Summarize Now** is a one-time action and will not affect the scheduled summary. Click **OK** to continue.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1321
Keysight, EX-1045
Page 1321

3. To verify summarization, navigate to **Log > View Log** in the left pane. Search for the message **Report Data Summarized** to verify that the Summarize Now action has completed.




4. When Summarize Now has completed, click the **UTM** tab at the top of the screen. In the left-most pane, click **MyReportsView** or click an appliance.

**Note** You may see incomplete data if you view the **Summary** section of a selected report before the **Summarize Now** process is complete. Wait for the **Report Data Summarized** message to be displayed in **Log > View Log**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1322
Keysight, EX-1045
Page 1322

5. In the center pane, click a report to expand it, then click the **Summary** option underneath it. For example, click **Bandwidth**, then click **Summary** to review the summarized bandwidth usage data.




| | Hour | Connections | Cost (USD) | MBytes | % of MBytes |
|---|---|---|---|---|---|
| 1 | 00:00 - 01:00 | 6031 | 1.903 | 3.807 | 3.6% |
| 2 | 01:00 - 02:00 | 6067 | 1.914 | 3.828 | 3.6% |
| 3 | 02:00 - 03:00 | 5920 | 1.877 | 3.753 | 3.6% |
| 4 | 03:00 - 04:00 | 5977 | 2.103 | 4.205 | 4.0% |
| 5 | 04:00 - 05:00 | 5994 | 1.915 | 3.831 | 3.6% |
| 6 | 05:00 - 06:00 | 6079 | 1.925 | 3.85 | 3.6% |

6. Navigate to the Summary section of other reports in the center pane to see other summarized data.

# Configuring the Syslog Deletion Schedule Settings

Syslog files sent from SonicWALL appliances are stored on the ViewPoint system, and are consolidated into the syslog database. The Summarizer processes the syslog data and stores the processed data in the summary database. After summarization and after the configured period of syslog storage, the syslog data can be periodically deleted from the system. This is necessary as the syslog files and database can consume a lot of space on the file system.

This section of the the Summarizer page also provides a way to delete summarized data for a certain date. For example, if summarized data is kept for a long time, such as 90 days, then you could use this option to remove some summarized data from a particular date within the 90 day period if the stored data was becoming too large.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1323
Keysight, EX-1045
Page 1323



**Tip** Run your database maintenance jobs soon after the completion of the scheduled tasks configured on this page for summarizing data and deleting old syslog data.

For information about setting the number of days to store syslog files, the syslog database, and the summary database, see the "Configuring Data Storage Settings" section on page 139.

ViewPoint requires large amounts of disk space for raw data storage. In previous versions, the maximum raw syslog database size was 2 GB. ViewPoint now provides enhanced database capacity by creating a new 2 GB database everyday. Each file name includes the date it was created for easy reference. Raw syslog data is used to create Custom Reports for UTM and SSL-VPN appliances.

To configure the syslog and summarized data deletion settings, perform the following:

1. On the **Console** panel, navigate to **Reports > Summarizer**.
2. Under **Syslog Deletion Schedule**, select the time for daily deletion in the hour and minute **Delete Syslog Data Daily at** drop-down lists. Syslog data will be deleted at this time only after being stored for the number of days configured.
3. Click the **Update** button to the right of this field.
4. To delete summarized data from a specific date, enter a date in the form mm/dd/yyyy in the **Delete Summarized Data For** field.
5. Click the **Update** button to the right of this field.

# Configuring Host Name Resolution

The Host Name Resolution feature allows the administrator to enable and configure the time period for the name resolution crawler. The name resolution crawler periodically resolves host names for IP addresses found in reporting data. Once the host name is resolved, the name will appear in place of the IP address in reports that contain it. Over time, more host names will appear in the report data as they are added to the list.

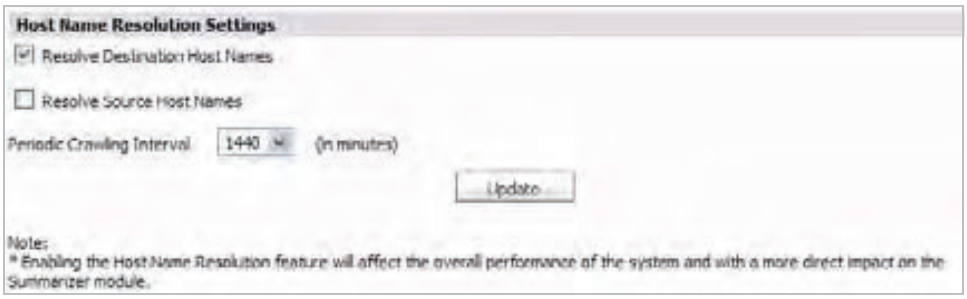
The name resolution crawler runs by default every 24 hours (1440 minutes) and can be configured to run every 1 to every 60 hours.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1324
Keysight, EX-1045
Page 1324

To use the Host Name Resolution feature, perform the following steps:

1. On the **Console** panel, navigate to **Reports > Summarizer**. The Host Name Resolution Settings section is displayed at the bottom of the page.




2. To resolve host names for destination IP addresses, select the **Resolve Destination Host Names** checkbox.
3. To resolve host names for source IP addresses, select the **Resolve Source Host Names** checkbox.
4. To set the interval at which the name resolution crawler runs, select the number of minutes in the **Periodic Crawling Interval** drop-down list.

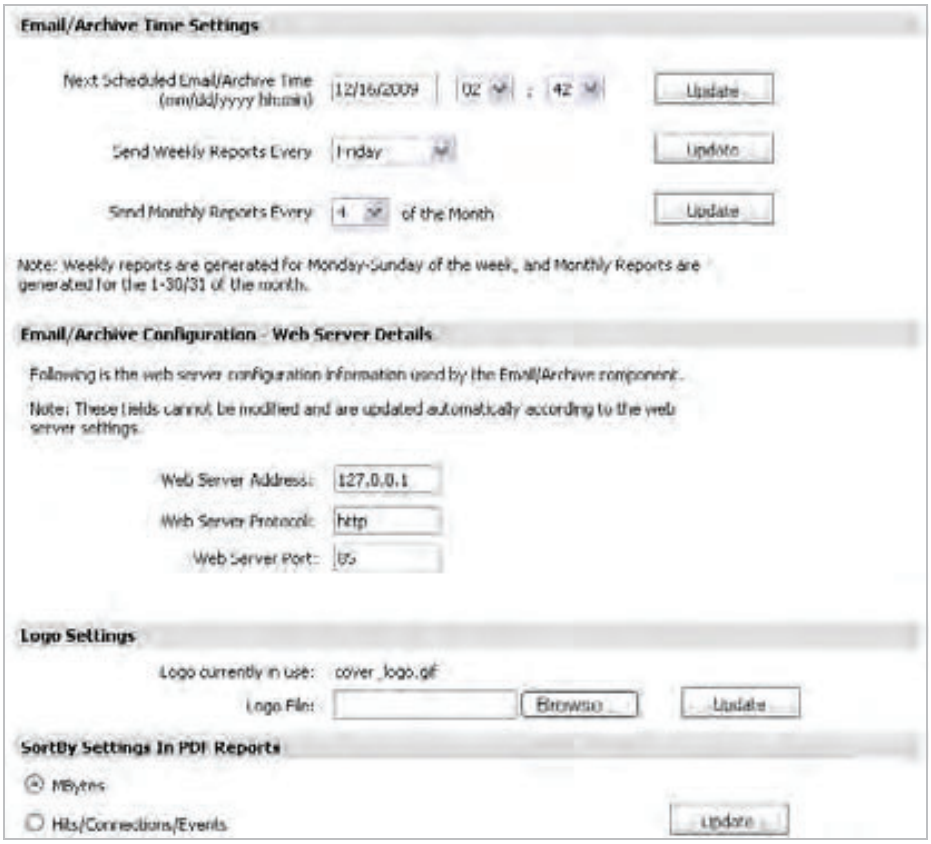
   Performance may be affected while the name resolution crawler is running, especially for the Summarizer module.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1325
Keysight, EX-1045
Page 1325

# Email/Archive

The Console > Reports > Email/Archive page provides global options for setting the time and interval for emailing/archiving scheduled reports, and global settings for the Web server, logo, and PDF sorting options.




# Configuring Email/Archive Settings

To configure Email/Archive and Web server settings, perform the following steps:

1. Click the **Console** tab, expand the **Reports** tree and click **Email/Archive**. The Email/Archive page displays.
2. To set the next archive time, enter the date and time in the **Next Scheduled Email/Archive Time** fields and click **Update**.
3. To specify the day to send weekly reports, select the day from the **Send Weekly Reports Every** list box and click **Update**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1326
Keysight, EX-1045
Page 1326

4. To specify the date to send monthly reports, select the date from the **Send Monthly Reports Every** list box and click **Update**.
5. If the Web server address, port, or protocol has changed since SonicWALL ViewPoint was installed, the new values will automatically appear in the **Email/Archive Configuration** section. These settings can be modified on the System Interface, and cannot be modified here.
6. Under Logo Settings, you can select a logo to be used on reports. By default, the SonicWALL logo is used. To select another logo, click **Browse** next to the **Logo File** field or type the path and filename into the field, and then click **Update**.
7. Under SortBy Settings for PDF Reports, select one of the following as the sorting criteria for reports and then click **Update**.
   - **Mbytes** - Sort reports by the number of megabytes in each entry
   - **Hits/Connections/Events** - Sort reports by the number of hits, connections, or events, depending on the type of report

# Scheduled Reports

The Scheduled Reports page allows you to manage all the report schedules in the system from a central location. This page lists all the schedules in the system, enabling you to monitor the status of these recurring schedules and re-send failed schedules, if needed. For information on adding a new scheduled report, see “Adding or Editing a Scheduled Report” section on page 135.

Under Search Results, the table indicates whether each schedule is enabled, along with information about the last execution time of a schedule, whether it ran successfully and the error that occurred if it failed, the last run type (scheduled or one time run), along with the node, owner and other relevant information.

The Summary section provides status information on your report schedules.

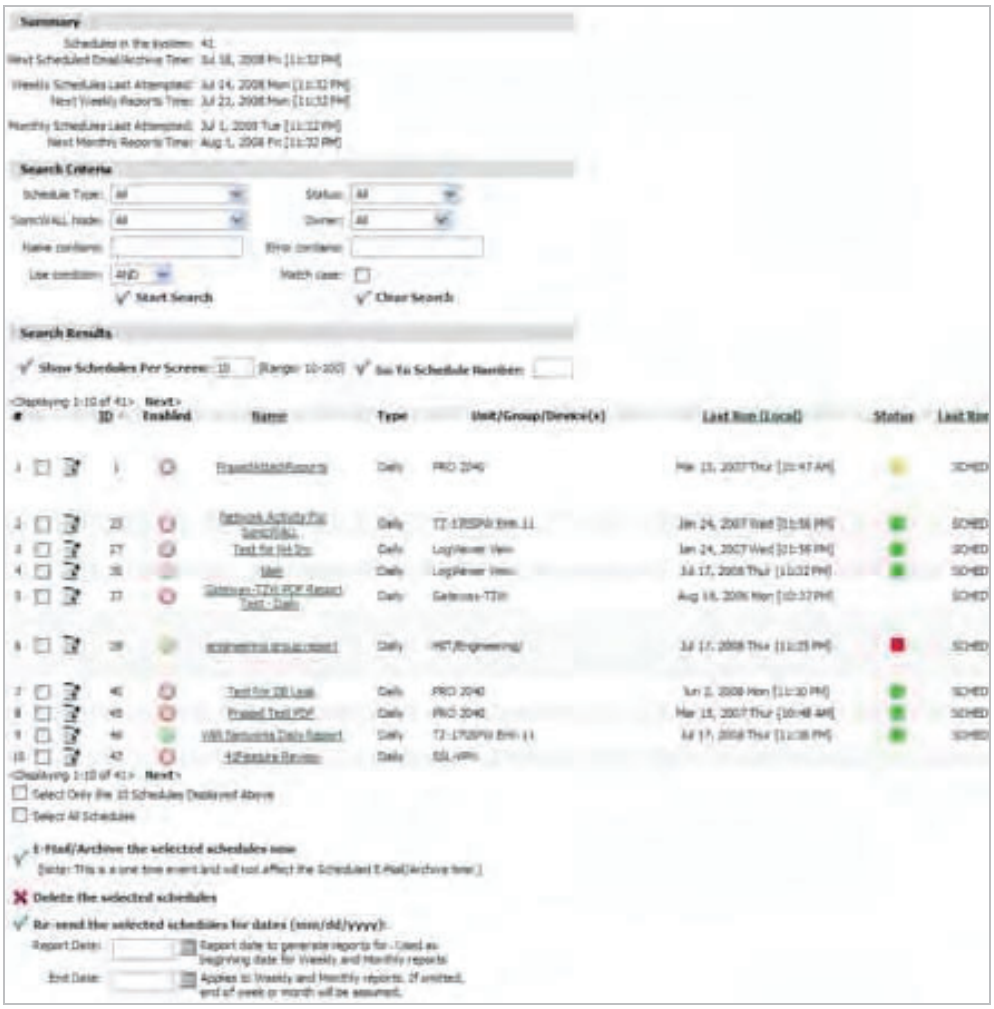
The Search Criteria section provides settings for searching report schedules. Results of your searches are displayed in the Search Results section.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1327
Keysight, EX-1045
Page 1327

**To search for scheduled reports:**

1. Click the **Console** tab, expand the **Reports** tree and click **Scheduled Reports**. The **Scheduled Reports** page displays.




2. Define the **Search Criteria** tab. The Search Criteria tab contains the following elements to refine your search:
   - **Schedule Type** - Select from the following schedule types:
     - –All Schedules
     - –Daily Schedules
     - –Weekly Schedules
     - –Monthly Schedules
   - **Status** - Select from the following status conditions:
     - –All



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1328
Keysight, EX-1045
Page 1328

  - Failed
  - In Progress
  - Success
  - In Queue
  - Partial Failure
- **SonicWALL Node** - Select from the following SonicWALL nodes:
  - All
  - Per Unit View
- **Owner** - Displays the owner (admin).
- **Name Contains** - Enter a context string to search by keywords.
- **Error Contains** - Enter a context string to search by keywords.
- **Use Condition** - Select from the following conditions:
  - And
  - Or
- **Match Case** - Select this checkbox to make your searches case sensitive.

3. Click **Start Search** to begin searching, or click **Clear Search** to reset all fields and start over.

The results of your search are displayed in a table in the Search Results section. You can adjust the number of schedules displayed, go directly to a row of the table, or navigate to other screens by clicking on links within the table.

**To work with the search results:**

1. To adjust the number of schedules displayed in the table, enter a number of rows to display in the **Show Schedules Per Screen** field, and then click on the checkmark.
2. To go directly to a row of the table, enter the row number in the **Go To Schedule Number** field, and then click on the checkmark.
3. The columns in the table are as follows:
   - The check box allows you select the schedule for emailing or archiving.
   - The notepad icon is a link to the Schedule Properties page.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1329
Keysight, EX-1045
Page 1329

- **ID** - The schedule ID number used to identify this schedule. You can click on the column heading to sort by this field. An arrow is displayed in the column heading when this field is the basis for sorting, and indicates ascending or descending order.
- **Enabled** - A green check mark indicates that this schedule is enabled, and a red X means that it is disabled.
- **Name** - The name of the report. Click on the highlighted report name link to access the report for editing. You can click on the column heading to sort by this field. An arrow is displayed in the column heading when this field is the basis for sorting, and indicates ascending or descending order.
- **Type** - All, Daily Schedules, Weekly Schedules, and Monthly Schedules.
- **Unit/Group/Devices(s)** - The host name of the SonicWALL appliance.
- **Last Run (Local)** - The date when the report was last generated. You can click on the column heading to sort by this field. An arrow is displayed in the column heading when this field is the basis for sorting, and indicates ascending or descending order.
- **Status** - Includes the following report status options:

  **–Blue**: Queued, waiting to be processed.

  **–Yellow**: Currently processing.

  **–Orange**: Report completed with errors.

  **–Red**: Report failed with errors.

  **–Green**: Report processed successfully.

  You can click on the column heading to sort by this field. An arrow is displayed in the column heading when this field is the basis for sorting, and indicates ascending or descending order.
- **Last Run Type -** Indicates if the most recent run was a scheduled run or a one-time execution. You can click on the column heading to sort by this field. An arrow is displayed in the column heading when this field is the basis for sorting, and indicates ascending or descending order.
- **Last Error** - Displays the error condition from the most recent run, if any. You can click on the column heading to sort by this field. An arrow is displayed in the column heading when this field is the basis for sorting, and indicates ascending or descending order.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1330
Keysight, EX-1045
Page 1330

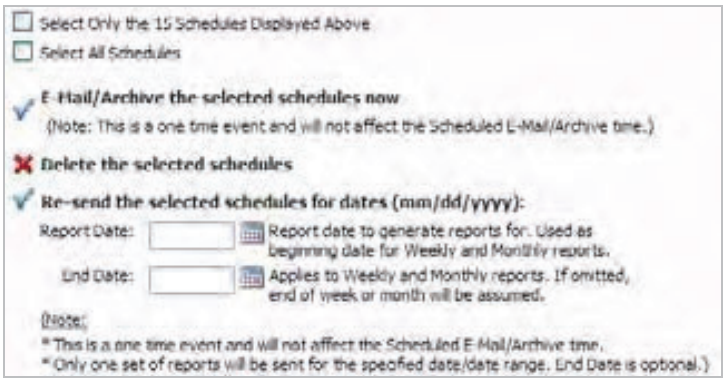
- **Owner** - Indicates the user ID of the user who created the schedule. You can click on the column heading to sort by this field. An arrow is displayed in the column heading when this field is the basis for sorting, and indicates ascending or descending order.

4. To view the properties for a schedule, click the notepad icon in that row. The Schedule Properties page displays.

5. To view the report, click on the name of the report. Your screen will change to the report screen on the UTM or SSL-VPN panel.

## Resending Schedules

Apart from selecting multiple schedules for a one-time execution by selecting the appropriate checkboxes and clicking the **Email/Archive the Selected Schedules now**, you can re-send required schedules using the **Re-send the selected schedules for dates** option.




To resend any schedules, follow the procedures below:

1. Select the Schedule Type (Daily, Weekly, or Monthly) from the Search Criteria section and click **Start Search**. This lists all the schedules of the selected type. Select the checkboxes of the schedules you want to resend.
2. Provide a start date (and an end date if applicable). Reports are generated for the specified date/date range.
3. Click **Re-send the selected schedules for dates**. Reports are generated for the specific dates and emailed/archived as a one time option for all the schedules selected.

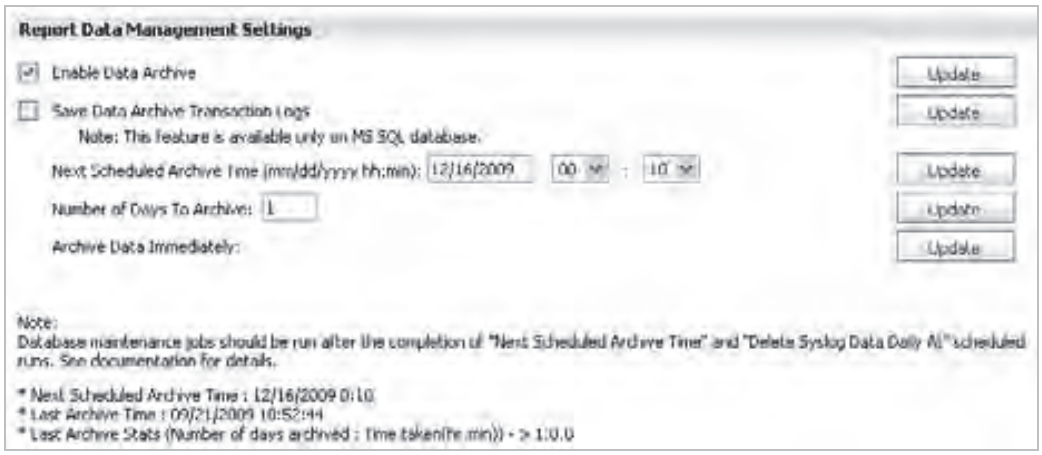


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1331
Keysight, EX-1045
Page 1331

# Management

Report Data Management allows the SonicWALL ViewPoint administrator to backup large amounts of report data incrementally and at specified intervals using MDTA. Typically, the total amount of data stored in an archive is equal to at least 30 days, although best benefits are seen when storing at least 60 days of summarizer data. MDTA allows this archive to be built over time, archiving as little as 1 day of data each time the MDTA process is run.

**Note** Total days to store summarized data in reports is set separately in the **Console > Reports > Summarizer** screen. Set this field for a value greater than 60 days for best results.




# Configuring Report Data Management

As an administrator, you choose the number of days worth of data to archive each time the MDTA process is run. With the exception of the current month, all available data is eligible for archiving. For example, if you specify 3 days as the number of days to archive, MDTA will archive 3 days of data, starting with the oldest available data and will repeat this process every day. In order to obtain optimal performance when viewing reports however, SonicWALL ViewPoint ensures that the current month is always kept in un-archived form.

**Step 1** In the ViewPoint Administrator Interface, navigate to **Console > Reports > Management**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1332
Keysight, EX-1045
Page 1332

**Step 2** Check the box next to **Enable Data Archive** and click the corresponding **Update** button.

**Step 3** Configure Data Archiving as follows, clicking the corresponding **Update** button after each line is completed:

| | |
|---|---|
| Save Data Archive Transaction Logs | Select to save truncated data archive transaction logs during each MDTA operation. Click the **Update** button. This option is deselected by default in order to conserve disk space. |
| Next Scheduled Archive Time | Schedule an initial date (mm/dd/yyyy) and time (in 24-hour format) for the MDTA operation. Click the **Update** button. MDTA operations will take place every day at the time you specify, starting with your initial date selection. |
| Number of Days to Archive | Specify the number of days worth of data to consider for each MDTA operation. |
| Archive Data Immediately | Press this button to immediately start an on-demand MDTA operation. The archive will run immediately but your scheduled archive operation will still take place. |



**Note** High-traffic systems can generate reports that consume large amounts of memory, disk space and CPU time when using MDTA. Set your **Number of Days to Archive** and **Scheduled Archive Time** accordingly. To view when MDTA operations are starting and how long the process is taking, navigate to the **Console > Log > View Log** screen and look (or search) for or “start” and “completed” times for “Report Data Archive.”



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1333
Keysight, EX-1045
Page 1333

# CHAPTER 9
# Using Diagnostics

This chapter describes the diagnostic information that ViewPoint provides, including summarizer status information.

This chapter includes the following sections:


- "Summarizer Status" section on page 90

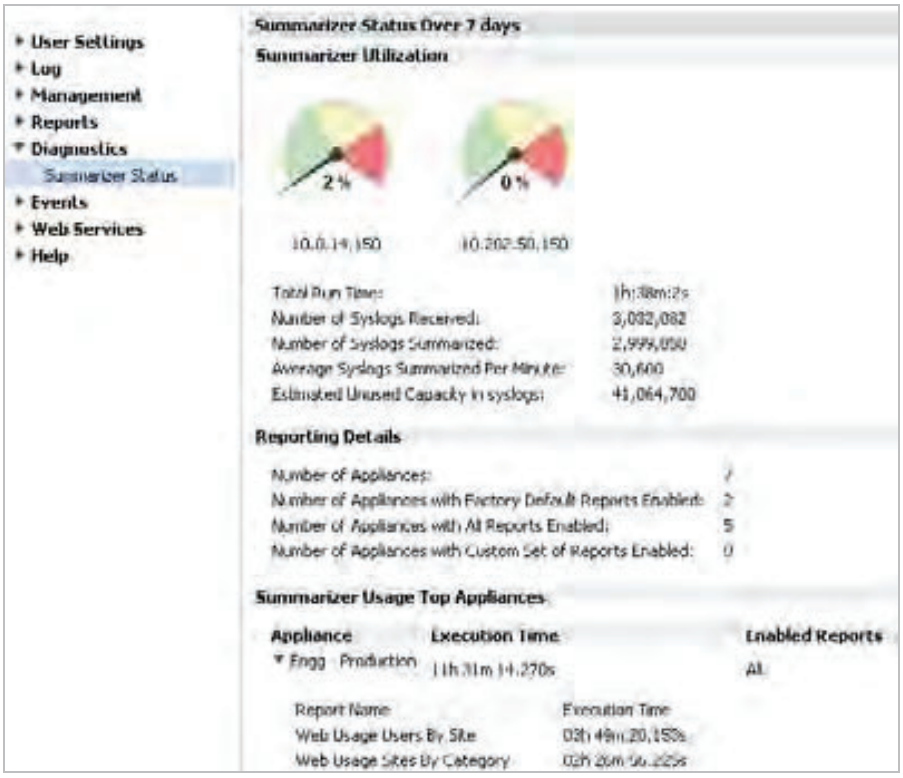




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1334
Keysight, EX-1045
Page 1334

# Summarizer Status

The **Summarizer Status** page displays overall summarizer utilization information for the deployment including database and syslog file statistics, and details on the current status of each summarizer.




The Summarizer Status screen provides performance metrics for your network administrator to plan, design, and expand your ViewPoint server deployment. This feature has information on the Syslog Collector and Summarizer metrics. The Summarizer metrics are available only for ViewPoint deployments that have Distributed Summarizer enabled (enabled by default on ViewPoint 5.1). The metrics are available for the past 24 hours, past seven days, and past 30 days.

These metrics are reset (to zero), every 24 hours for daily metrics, every seven days for weekly metrics, and every 30 days for monthly metrics. Weekly metrics are not shown unless the data collection for weekly metrics started earlier than the daily metrics. Similarly, monthly metrics are not shown unless data collection for monthly metrics started earlier than for daily and weekly metrics. ViewPoint will not display metrics for a component if the daily statistics collection started more than 26 hours earlier. This generally indicates that the component is not active.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1335
Keysight, EX-1045
Page 1335

You can receive alert emails when Summarizer Status shows any abnormalities.

To reach the Summarizer Status screen, navigate to the **Console** panel of ViewPoint and then to **Diagnostics > Summarizer Status**.

The Summarizer Status page is divided into a section showing the overall deployment-wide summarizer status and sections with details for each summarizer. See the following sections:


- Summarizer Status Over 7 Days, page 91
- Details for Summarizer at <IP Address>, page 93


# Summarizer Status Over 7 Days

The Summarizer Status Over 7 Days section displays overall summarizer utilization information for the deployment including database and syslog file statistics. Results are calculated over the last 7 days, with historical data available over the last 30 days.

## Summarizer Utilization

The top Summarizer Utilization section shows the average utilization of the summarizer over the applicable time period. The Dial Charts show the percent of total capacity used by the Syslog Collector or the Summarizer. The following metrics are also displayed in the Summarizer Utilization section:

**Total Run Time**: Total amount of time spent generating summarization statistical data and results over the applicable time period.

**Number of Syslogs Received**: Total number of syslogs received by the Summarizer over the applicable time period.



**Note** Not all syslogs are summarized – some syslogs, such as “heartbeat messages” are ignored. When **Web Event Consolidation/Home Port Reporting** is enabled, several syslogs may be ignored or alternatively, consolidated into a single syslog. If your appliance is managed by a different Agent, the results are not summarized here.

**Number of Syslogs Summarized**: Total number of syslogs summarized over the applicable time period.

**Average Syslogs Summarizer per Minute**: Average number of syslogs summarized per minute over the applicable time period.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1336
Keysight, EX-1045
Page 1336

**Estimated Unused Capacity in Syslogs**: The estimated remaining capacity of the summarizer in terms of the number of syslogs it can summarize, based on the time taken and number of syslogs summarized over the applicable time period. This number does not include the discarded syslogs.



**Tip** Usage Example: For this example, let's assume that the syslogs summarized per minute on a system is **18,108**, and the average number of syslogs received on that system is **91** per firewall, per minute. Divide the number of syslogs per minute (18,108) by the number of syslogs per appliance per minute (91). This yields an estimate of **198** security appliances, assuming that the current appliances are a fair sample of the security appliances on your network.
This simple math gives a reasonable estimate of the total number of security appliances this system should be able to handle, assuming that the Summarizer was to constantly summarize 24 hours (as in the case of a dedicated Summarizer).

## Reporting Details

The Reporting Details section shows the number of appliances in the deployment, and the number with the following types of reports enabled:

- Factory default reports
- All reports
- Custom set of reports

## Summarizer Usage Top Appliances

The Summarizer Usage Top Appliances section displays information about the appliances in the deployment that used the most summarizer time. Details are given about which reports were generated and their summarizer execution time.

## Database Statistics

The size is displayed for each of the following databases:

- Current
- Archive
- Raw Syslog



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1337
Keysight, EX-1045
Page 1337

### Syslog File Storage Statistics

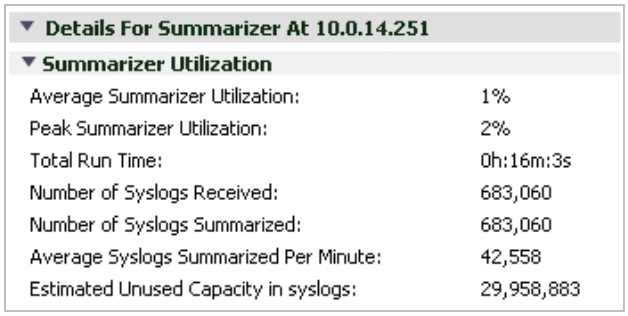
The size is displayed for each of the following syslog directories:

- Current
- Archived
- Bad

## Details for Summarizer at <IP Address>

### Summarizer Utilization

The Summarizer Utilization section for a specific summarizer shows the same information described above for the entire deployment, but only shows the values for this summarizer.



| Details For Summarizer At 10.0.14.251 | |
|---|---|
| **Summarizer Utilization** | |
| Average Summarizer Utilization: | 1% |
| Peak Summarizer Utilization: | 2% |
| Total Run Time: | 0h:16m:3s |
| Number of Syslogs Received: | 683,060 |
| Number of Syslogs Summarized: | 683,060 |
| Average Syslogs Summarized Per Minute: | 42,558 |
| Estimated Unused Capacity in syslogs: | 29,958,883 |

### Reporting Details

The Reporting Details section shows the number of appliances serviced by this summarizer, and the number with the following types of reports enabled:

- Factory default reports
- All reports
- Custom set of reports

### Summarizer Usage Top Appliances

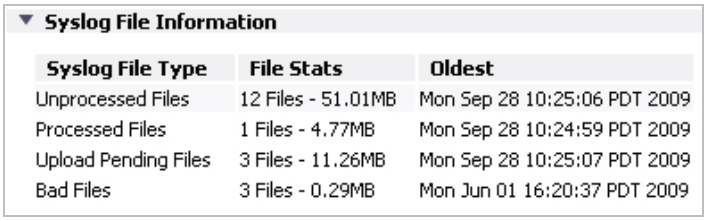
The Summarizer Usage Top Appliances section displays information about the appliances serviced by this summarizer that used the most summarizer time. Details are given about which reports were generated and their summarizer execution time.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1338
Keysight, EX-1045
Page 1338

## Syslog File Information

This section displays syslog file details for the selected summarizer.



Syslog File Information

| Syslog File Type | File Stats | Oldest |
|---|---|---|
| Unprocessed Files | 12 Files - 51.01MB | Mon Sep 28 10:25:06 PDT 2009 |
| Processed Files | 1 Files - 4.77MB | Mon Sep 28 10:24:59 PDT 2009 |
| Upload Pending Files | 3 Files - 11.26MB | Mon Sep 28 10:25:07 PDT 2009 |
| Bad Files | 3 Files - 0.29MB | Mon Jun 01 16:20:37 PDT 2009 |

The Syslog File Information table is divided into three columns:

- **Syslog File Type**: The type of files being reported on.

  There are ten main syslog file types:

  - Processed Files
  - Unprocessed Files
  - Grouped Files
  - Not Mine Files
  - Infected Files
  - Archived Files
  - Bad Files
  - Upload Pending Files
  - Uploaded Files
  - Bad Upload Files
- **File Stats**: The number of syslog files in the category and their size in Megabytes.
- **Oldest**: The date and time on the oldest file in the category.

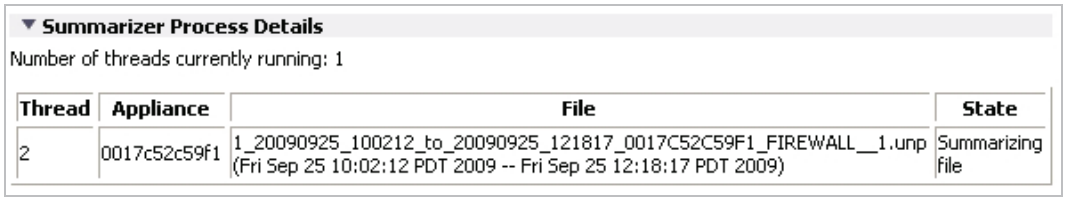
## Summarizer Process Details

The Summarizer Process Details section shows what tasks the summarizer is performing at the moment the **Console > Diagnostics > Summarizer Status** page displays. Refresh your browser display or leave the page and return to it to update the information.

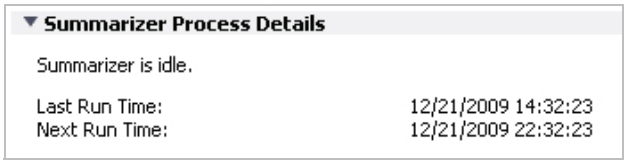


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1339
Keysight, EX-1045
Page 1339

If the summarizer is currently running, the page displays the thread, appliance identifier, file being used, and state of the summarizer.




If the summarizer is currently idle, the page displays the last run time and next run time.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1340
Keysight, EX-1045
Page 1340

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1341
Keysight, EX-1045
Page 1341

# CHAPTER 10
# Granular Event Management

This chapter describes how to configure and use the Granular Event Management (GEM) feature in a ViewPoint environment.

This chapter contains the following sections:


- "Granular Event Management Overview" section on page 97
- "Using Granular Event Management" section on page 99
- "Configuring Granular Event Management" section on page 101
- "Viewing Current Alerts" section on page 108


## Granular Event Management Overview

Granular Event Management (GEM) provides a customized and controlled manner in which events are managed and alerts are customized and enabled. On the Console panel, GEM allows you to systematically configure each sub-component of your alert in order for the alert to best accommodate your needs.

The GEM alert has multiple sub-components, some of which have further subcomponents. It is not necessary to configure all sub-components prior to creating an alert.

- **Severities**: Severity is used to tag an alert as Critical, Warning, or Information. Severities are included within each Threshold. You can change the severity levels of the threshold elements listed on the Console > Events > Threshold page.




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1342
Keysight, EX-1045
Page 1342

- **Thresholds**: A threshold defines the condition that must be matched to trigger an event and send an alert. Each threshold is associated with a Severity to tag the generated alert as critical, warning, or information.

  One or more threshold elements are defined within a threshold. Each threshold includes the following elements: an Operator, a Value, and a Severity. When a value is received for an alert type, the GEM framework examines threshold elements to find a match for the specified condition. If a match is found (one or more conditions match), the threshold with the highest severity containing a matching element is used to trigger an event.
- **Schedules**: You can use Schedules to specify the day(s) and time (intervals) in which to generate an alert. You can also invert a schedule, which means that the schedule is the opposite of the time specified in it. For example:
  - Generate an alert during weekdays only, or weekends only, or only during business hours.
  - Do not generate an alert during a time period when the unit, network, or database are down for maintenance.

# What is Granular Event Management?

The purpose of Granular Event Management is to provide all the event handling and alerting functionality for ViewPoint. The ViewPoint management interface provides screens for centralized event management on the Console panel, including screens for Events > Threshold, Schedule, and Alert Settings. The panel also provides an Events > Alert Settings screen where you can enable or disable alerts.

You can enable or disable an alert at the global or unit level in ViewPoint. At the global level, the alert is then applied to all units. Whenever you add a new unit to ViewPoint, the alerts set at the global level are applied to the new unit.

# How Does Granular Event Management Work?

The Granular Event Management framework provides customized event handlingfor specific alerts about database and database log size, and security service subscription licenses. For a list of the predefined alerts, see “Using Granular Event Management” on page 99.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1343
Keysight, EX-1045
Page 1343

# Using Granular Event Management

For convenience and usability, a number of default settings are predefined for severities, schedules, thresholds, and alerts. You can edit the predefined values to customize the settings for thresholds and schedules. The predefined defaults for each panel and screen are as follows:

***Table 1 GEM Predefined Default Objects***

| Panel | Screens | Predefined Default Objects |
|---|---|---|
| **Console** | **Events > Thresholds** | Unit Status |
| | | Database Size Status |
| | | Database Log Size Status (on MySQL DB only) |
| | | Summarizer Utilization |
| | | Summarizer Backed-Up Files |
| **Console** | **Events > Schedule** | **Schedule Groups:** |
| | | • 24x7 |
| | | • Weekdays 24 hours |
| | | • 8x5 |
| | | • Weekend |
| | | **Schedules:** |
| | | • Schedule: admin |
| | | • Monday 24 hours |
| | | • Monday business hours |
| | | • Tuesday 24 hours |
| | | • Tuesday business hours |
| | | • Wednesday 24 hours |
| | | • Wednesday business hours |
| | | • Thursday 24 hours |
| | | • Thursday business hours |
| | | • Friday 24 hours |
| | | • Friday business hours |
| | | • Saturday 24 hours |
| | | • Sunday 24 hours |



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1344
Keysight, EX-1045
Page 1344

| Panel | Screens | Predefined Default Objects |
|---|---|---|
| **Console** | **Events > Alert Settings** | Database Info |
| | | Database Size Status |
| | | Database Log Size Status (on MySQL DB only) |
| | | Summarizer Utilization Status |
| | | Summarizer Backed-Up Files Status (on MySQL DB only) |

# About Alerts

The **Events > Alert Settings** screens are available in the Console and UTM panels. You can enable or disable alerts on these screens.

The GEM framework provides different types of alert types for the respective areas of the ViewPoint application:

- UTM  panel: Alert settings for Reporting
- Console panel: Alert settings for the ViewPoint application

***Table 2 GEM Alert Types***

| Panel location | Available Alert Types |
|---|---|
| **Console** | Date Base Info |
| | Database Size Status |
| | Database Log Size Status (on MySQL DB only) |
| | Summarizer Utilization Status |
| | Summarizer Backed-Up Files Status (on MySQL DB only) |
| **UTM** | Anti Virus License |
| | CFS License |
| | Warranty License |
| | Anti Spyware License |
| | Intrusion License |



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1345
Keysight, EX-1045
Page 1345

# Configuring Granular Event Management

To set up the GEM environment after installing ViewPoint, start with the Events screens on the Console panel. You should examine the Threshold and Schedule screens and make any necessary configuration changes. Then you can enable alerts in the Events screens on the Console panel and UTM panel.

See the following sections:

- "Configuring Events on the Console Panel" section on page 101
- "Enabling or Disabling Alerts on the UTM Panel" section on page 107

## Configuring Events on the Console Panel

In the Events screens on the Console panel, you can configure the frequency of subscription expiration and task failure notifications, as well as severities, thresholds, schedules, and alerts for handling events.

See the following sections:

- "Configuring Event Thresholds" on page 101
- "Configuring Event Schedules" on page 104
- "Enabling or Disabling Alerts on the Console Panel" on page 107

### Configuring Event Thresholds

In the **Events > Threshold** screen, you can view existing event thresholds and configure their elements, and add custom thresholds. A threshold defines the condition for which an event is triggered. Predefined thresholds have names similar to predefined Alert Types. Each threshold can contain one or more threshold elements. An element consists of an Operator, a Value, and a Severity.

The following tasks are described in this section:

- "Editing an Event Threshold Element" on page 102
- "Enabling/Disabling Event Thresholds and Threshold Elements" on page 103

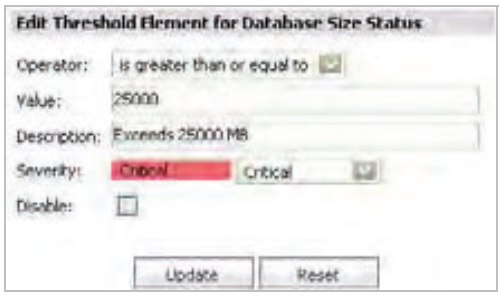


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1346
Keysight, EX-1045
Page 1346

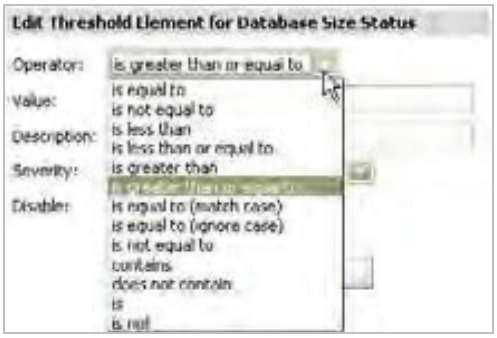
## Editing an Event Threshold Element

To edit an existing element of a Threshold, perform the following steps:

1. On the **Events > Threshold** screen, click the **Edit** icon located in the Configure column in the element row.
2. In the Edit Threshold Element window, you can edit the following fields:
   - Operator
   - Value
   - Description
   - Severity
   - Disable




3. In the **Operator** field, select from the drop down menu the type of operator to apply to your threshold element..

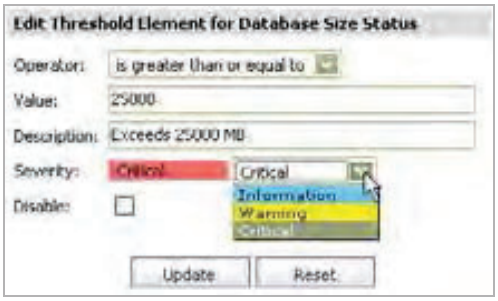



4. In the **Value** field, enter the value for your threshold element.
5. In the **Description** field, enter the description for your threshold element.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1347
Keysight, EX-1045
Page 1347

6. In the **Severity** field, select the severity priority from the drop down menu. These are color coded for your easy reference on the Events > Threshold screen.




7. To disable the threshold element, click the **Disable** check box. See "Enabling/Disabling Event Thresholds and Threshold Elements" section on page 103.
8. Click **Update**.

## Enabling/Disabling Event Thresholds and Threshold Elements

The GEM feature provides a **Disable** check box that allows you to disable or enable thresholds or individual elements within that threshold. If it is needed again, you can simply enable it.

You can disable a threshold by disabling all its elements. You can also disable individual elements within a threshold.

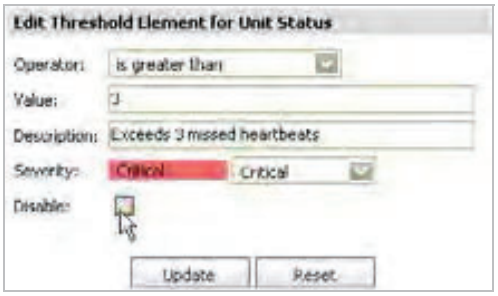
To enable or disable Thresholds and/or their elements, perform the following tasks:

1. On the **Console** panel, navigate to the **Events > Threshold** screen. On this screen, you are able to view existing Thresholds. You can also view existing elements within those thresholds by clicking the expand button by a threshold. You have the following two options for the enabling/disabling feature:
   - You can enable or disable a Threshold by disabling/enabling all the elements that exist within it.
   - You can enable/disable the individual elements within a Threshold.
2. To enable or disable a threshold and/or elements, click the edit button  that is on the element level.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1348
Keysight, EX-1045
Page 1348

3. Select the **Disable** checkbox to disable the element or de-select the **Disable** checkbox to enable the element.




4. Click **Update**.

## Configuring Event Schedules

The next component on the Console panel is **Events > Schedule**. In this screen, you can add, delete, or configure schedules and schedule groups.

Schedule groups are one or more schedules grouped within an object. Administrators and Owners can edit these objects. Other users should be able to view or use them only if the **Visible to Non-Administrators** check box is selected.

The following tasks are described in this section:


- "Adding an Event Schedule" on page 104
- "Editing an Event Schedule" on page 106
- "Adding an Event Schedule Group" on page 106
- "Deleting a Schedule or Schedule Group" on page 107


### Adding an Event Schedule

In **Events > Schedules** you can add, delete, or configure schedules. You will see your schedules and schedule groups, their descriptions, and whether they are enabled. You can also individually delete one schedule or schedule group at a time by selecting the trash-icon on the right hand side for each row. For quick reference, you can hover your mouse over the descriptions to quickly view the type of schedule and the days and times when it is active.

To add an event schedule, perform the following steps:

1. On the **Events > Schedules** screen, click **Add Schedule**.
2. Select the **Visible to Non-Administrators** check box if you want the schedule to be visible and usable by non-administrators.
3. To temporarily disable a schedule, select the **Disable** checkbox.

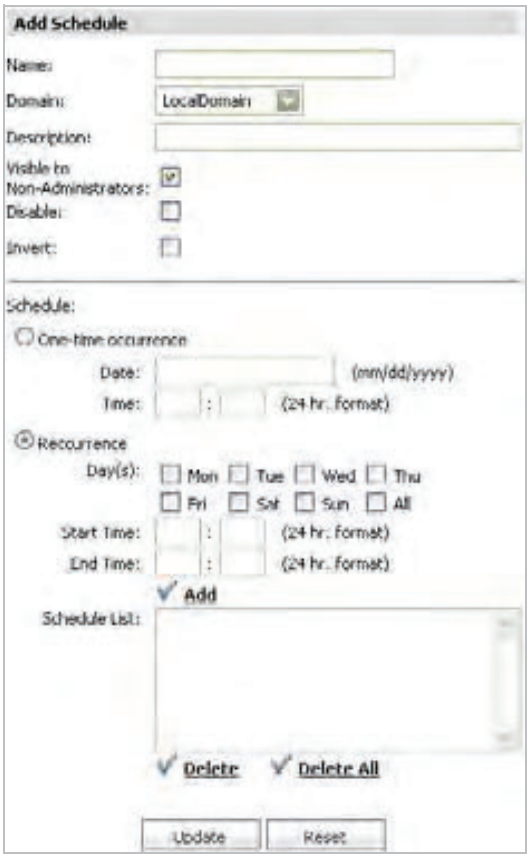


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1349
Keysight, EX-1045
Page 1349

4. Click **Invert** to create a schedule that is “off” during the dates and times that you specify.
5. In the Schedule field, you can create one or more schedules. For each schedule, configure either:

- **One Time Occurrence**
  - Fill in the **Date** and **Time** fields.
- **Recurrence**
  - Fill in **Days**, **Start Time**, and **End Time** fields.

6. Click **Add** to add this schedule to the **Schedule List** text box.




7. To delete an entry from the Schedule List text box, select the entry that you want to delete, and then click **Delete**. Click **Delete All** to delete all entries.
8. Click **Update** when you are finished.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1350
Keysight, EX-1045
Page 1350

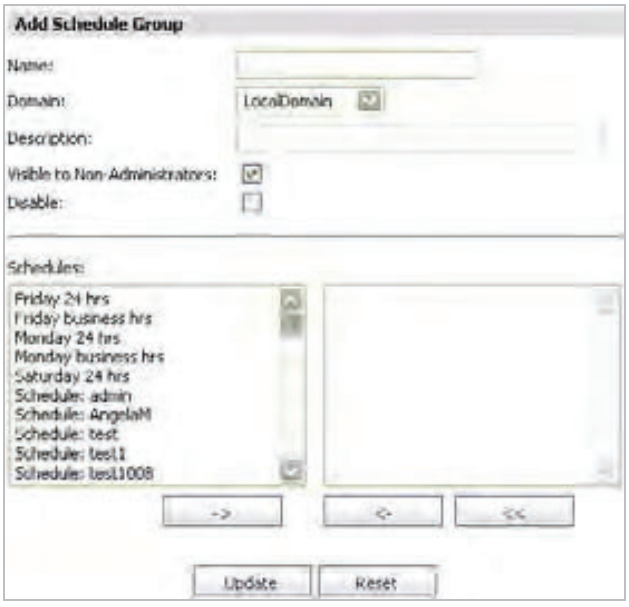
## Editing an Event Schedule

To edit an existing schedule, click the **Edit** icon on the right side of the **Events > Schedule** screen. The screen and procedure for editing are the same as those for adding a schedule. See "Adding an Event Schedule" section on page 104.

## Adding an Event Schedule Group

You can combine several schedules into a schedule group on the **Events > Schedule** screen. To add a schedule group, perform the following steps:

1. On the **Events > Schedule** screen, click the **Add Schedule Group** button.
2. Enter the name of your schedule group in the **Name** field.
3. Enter a description of your schedule group in the **Description** field.
4. Click the **Visible to Non-Administrators** check box to allow this schedule group to be viewed and used by non administrators.
5. Click the **Disable** check box to temporarily disable the schedule group.
6. In the **Schedules** field, select the schedule(s) to add to your schedule group, and then use the arrow buttons to move the selected schedule into or out of the group. To move multiple schedule groups and/or schedules all at once, hold the CTRL button on your keyboard while making your selections.




7. Click **Update**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1351
Keysight, EX-1045
Page 1351

### Deleting a Schedule or Schedule Group

You can delete schedules or schedule groups, or you can remove schedules from schedule groups.

To delete an event schedule, schedule group, or remove a schedule from a schedule group:

1. Navigate to the **Events > Schedule** screen.
2. Click the check boxes of the schedule groups or schedules that you want deleted. When you click the schedule group check box, the schedules within that schedule group will be deleted as well.
3. To remove a schedule from a schedule group, click the expand button on the schedule group, and select the schedules you wish to remove within that group.
4. To delete the selected schedule group(s) or remove the selected schedules from a group, click the **Delete Schedule Group(s)/Remove Schedules from Group** button.
5. To delete the selected schedule(s), click the **Delete Schedule(s)** button.

## Enabling or Disabling Alerts on the Console Panel

The **Console > Events > Alert Settings** screen provides predefined alerts that apply to ViewPoint as a whole. You can hover your mouse over these to display information about them.  You can enable or disable these alerts by selecting or clearing the checkbox in the **Enable** column for the alert.

# Enabling or Disabling Alerts on the UTM Panel

You can enable or disable alerts for events pertaining to security services licenses on the UTM panel.To enable or disable an alert:

1. To enable an alert, select the checkbox under **Enabled** in the row for the alert.
2. To disable an alert, clear the checkbox under **Enabled** in the row for the alert.
3. Click **Enable/Disable Alert(s)**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1352
Keysight, EX-1045
Page 1352

# Viewing Current Alerts

You can view a list of current alerts on the **Events > Current Alerts** page of the panel. Select a global view or unit to view current alerts for your selection.



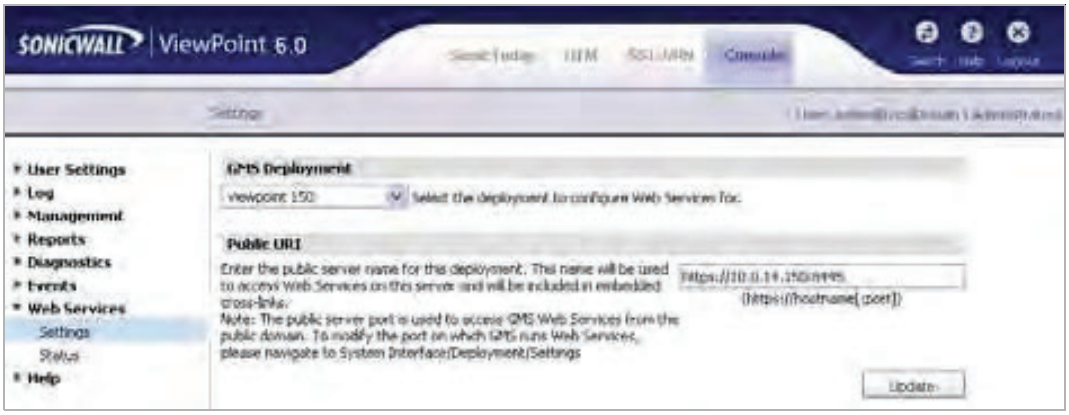



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1353
Keysight, EX-1045
Page 1353

# CHAPTER 11
# Web Services

This chapter provides information about the Web Services feature. Web Services is a software system designed to support interoperability between ViewPoint and other network appliances, servers, and devices through an application programming interface (API).

Web Services is located in the Console panel of the ViewPoint management interface:




This chapter includes the following sections:


- "URI Basics" section on page 110
- "Settings" section on page 111
- "Status" section on page 112



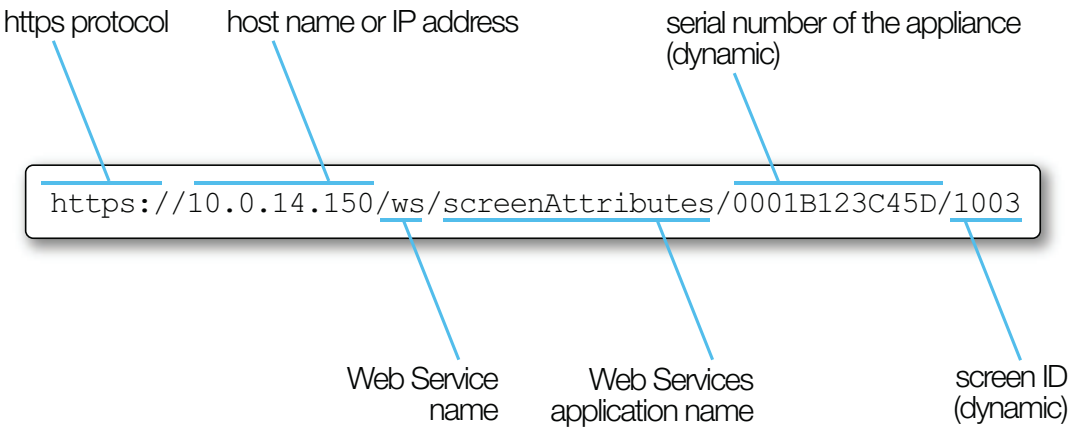


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1354
Keysight, EX-1045
Page 1354

# URI Basics

The URI is a HTTPS string which is used to identify Web Services resources. Each URI is composed of both static and dynamic parts which differ based on each particular deployment.

The following provides a typical, though not comprehensive, URI example:




**Note** For more information on configuring and using Web Services in your deployment, download the *GMS Web Services Technote* at: <http://www.sonicwall.com/us/support.html>



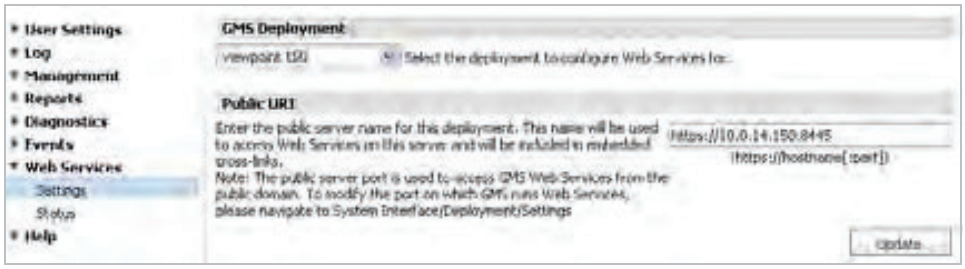
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1355
Keysight, EX-1045
Page 1355

# Settings

The Settings screen allows configuration of a secure HTTPS Public URI for use with Web Services features. The public URI specified here is used to access Web Services and to ensure proper embedded cross-links between Web Services applications.

To configure Web Services Settings:

1. Navigate to the **Web Services > Settings** screen on the Console panel.




2. Choose which deployment you wish to configure from the drop-down list in the **GMS Deployment** section.
3. Enter the public server name and port in the **Public URI** section. This field is typically pre-populated during the ViewPoint install/setup process.
4. Click the **Update** button to save your changes.



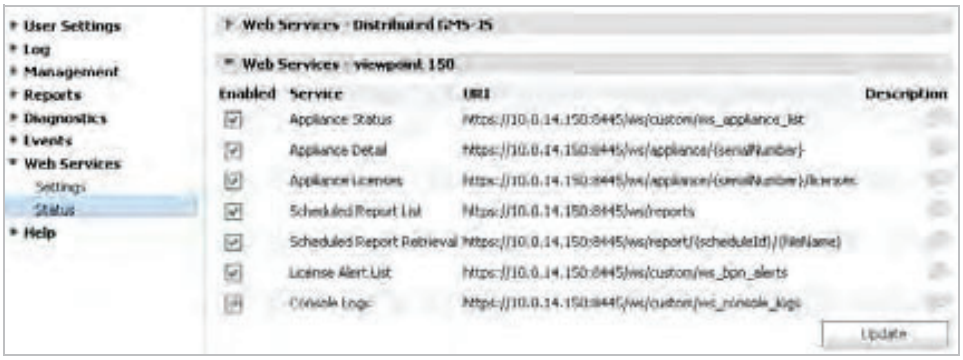
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1356
Keysight, EX-1045
Page 1356

# Status

The status screen allows the administrator to view, enable, and disable individual Web Services across one or more ViewPoint deployments.

To view and configure Web Services status:

1. Navigate to the **Web Services > Status** screen on the Console panel.




2. Select or deselect the **Enabled** checkbox for the service(s) you wish to enable or disable.
3. Click the **Update** button to save your changes.
4. The Web Services table, in the **Web Services > Status** screen gives the following information about each Web Service:

| Feature | Description |
|---|---|
| Enabled | If selected, this feature is currently enabled |
| Service | Indicates the name of the Web Service |
| URI | Indicates the full URI used to access this Web Service |
| Description | Provides a description of the Web Service |



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1357
Keysight, EX-1045
Page 1357

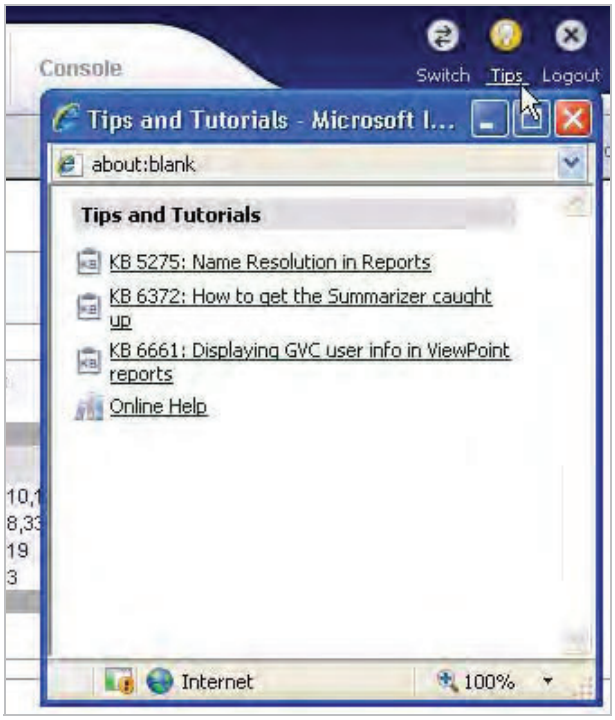
# CHAPTER 12
# Using ViewPoint Help

To access the ViewPoint online help, click the blue help button 

in the top-right corner of the ViewPoint user interface.

## Tips and Tutorials

Tips and tutorials are available in some pages of the user interface, and are denoted by a “Lightbulb” icon:






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1358
Keysight, EX-1045
Page 1358

To access tips and tutorials:

1. Navigate to the page where you need help.
2. If available, click the Lightbulb icon in the upper right-hand corner of the window. Tips, tutorials, and online help are displayed for this topic.

# About ViewPoint

The **Console > Help > About** page displays the version of ViewPoint being run, who the ViewPoint is licensed to, database information, and the serial number of the ViewPoint.

To access the ViewPoint online help, click the blue help button 

in the top-right corner of the ViewPoint user interface.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1359
Keysight, EX-1045
Page 1359

# CHAPTER 13
# ViewPoint Reporting Features

This chapter describes how to use ViewPoint reporting, including the type of information that can appear in reports. A description of the available features in the user interface is provided. Settings for reporting on the Console panel are described.

This chapter includes the following sections:


- "ViewPoint Reporting Overview" section on page 115
- "Navigating ViewPoint Reporting" section on page 119
- "Showing Domain Names in Reports" section on page 130
- "Managing ViewPoint Reports on the Console Panel" section on page 131


# ViewPoint Reporting Overview

Monitoring critical network events and activity, such as security threats, inappropriate Web use, and bandwidth levels, is an essential component of network security. ViewPoint Reporting complements SonicWALL's Internet security offerings by providing detailed and comprehensive reports of network activity.

The ViewPoint Reporting Module is a software application that creates dynamic, Web-based network reports. The ViewPoint Reporting Module generates both real-time and historical reports to offer a complete view of all activity through SonicWALL Internet security appliances. With ViewPoint Reporting, you can monitor network access, enhance security, and anticipate future bandwidth needs.




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1360
Keysight, EX-1045
Page 1360

You can search saved reports by using the report search bar, available in most report screens in the ViewPoint UI. The search bar provides pre-populated quick settings for the search field, and a drop-down calendar for the start and end dates. The search operator field offers a comprehensive list of search operators that varies depending on the search field, which can be either text-based or numeric.

You can search all columns of report data except columns that contain computed values, such as %, Cost, or Browse Time. ViewPoint waits until you click Search before it begins building the new report.

The ViewPoint Reporting Module:

- Displays bandwidth use by IP address and service
- Identifies inappropriate Web use
- Provides detailed reports of attacks
- Collects and aggregates system and network errors
- Shows VPN events and problems
- Tracks Web usage by users and by Web sites visited
- Provides detailed daily firewall logs to analyze specific events.



**Note** The ViewPoint Reporting Module receives its information from the stream of syslog data sent by each SonicWALL appliance and stores it in the SonicWALL ViewPoint database or as files on the hard-disk.



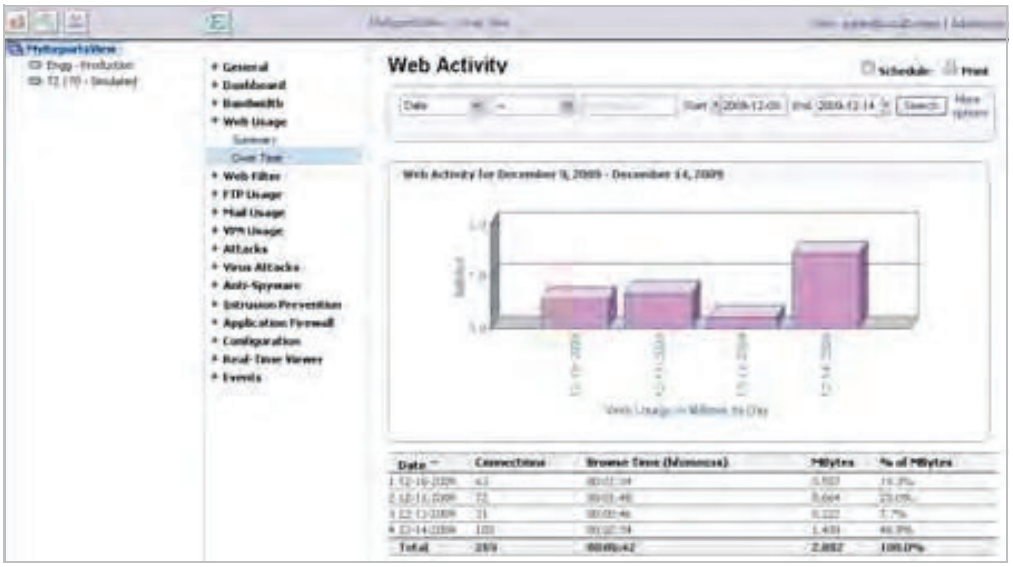
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1361
Keysight, EX-1045
Page 1361

# Viewing ViewPoint Reports

The ViewPoint reports are available on the **UTM** and **SSL-VPN** tabs of the ViewPoint interface:




The ViewPoint Reports view is divided into three panes:




- A list of individual units referred to as the TreeControl: In the left pane, you can select the top level view or a unit to display reports that apply to the selected view or unit. The top level view is **MyReportsView**.
- A list of reports: The middle pane provides a list of available reports that changes according to your selection in the TreeControl pane. The reports are divided into categories. You can click on the plus sign next to a category to view the list of reports in that category. You can click on an individual report name to view that report.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1362
Keysight, EX-1045
Page 1362

- The report: The right pane displays the report that you selected in the middle pane for the view or unit that you selected in the TreeControl. For most reports, the search bar is provided at the top of the pane. Above the search bar a link to the Scheduler is provided. You can change the time for the report to run by clicking the **Schedule** link or its clock icon in the upper right. A quick access link to your system's printer is also available in the upper right corner. To print the report, click the **Print** link or icon. To access the display settings for the report, click **More Options** to the right of the search bar.

The SonicWALL ViewPoint reporting feature provides the following configurable reports:

***Table 3 Configurable Reports***

| | |
|---|---|
| **General** | Provides general unit and license status. |
| **Dashboard** | Provides a high-level activity summary. |
| **Custom Report*** | Provides Internet Activity and Website Filtering reports with details from raw data<br>*Custom Reports are only available at the unit level. |
| **Bandwidth** | Provides bandwidth usage reports. |
| **Services*** | Provides events and usage by service protocol.<br>*Services reporting is only available at the unit level. |
| **Web Usage** | Provides Web usage reports. |
| **Web Filter** | Provides web filter event reports. |
| **FTP Usage** | Provides FTP usage reports. |
| **Mail Usage** | Provides mail usage reports. |
| **VPN Usage** | Provides VPN usage reports. |
| **Attacks** | Provides attack event reports. |
| **Virus Attacks** | Provides virus attack event reports. |
| **Anti-Spyware** | Provides spyware event reports. |
| **Intrusion Prevention** | Provides intrusion event reports. |
| **Application Firewall** | Provides Application Firewall reports. |
| **Authentication** | Provides login reports. |



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1363
Keysight, EX-1045
Page 1363

# Navigating ViewPoint Reporting

ViewPoint Reporting is a robust and powerful tool you can use to view detailed reports for individual SonicWALL appliances.

This section describes each view and what to consider when making changes. It also describes the Search Bar and display options for interactive reports, as well as other enhancements provided in SonicWALL ViewPoint . See the following sections:


- “Global Views” on page 120
- “Unit View” on page 121
- “Using Interactive Reports” on page 122
- “Searching for a Report” on page 123
- “Collapsible TreeControl Pane” on page 128
- “Enabling/Disabling Scheduled Reports” on page 128
- “Combined Reports” on page 128
- “Improved Navigation” on page 129

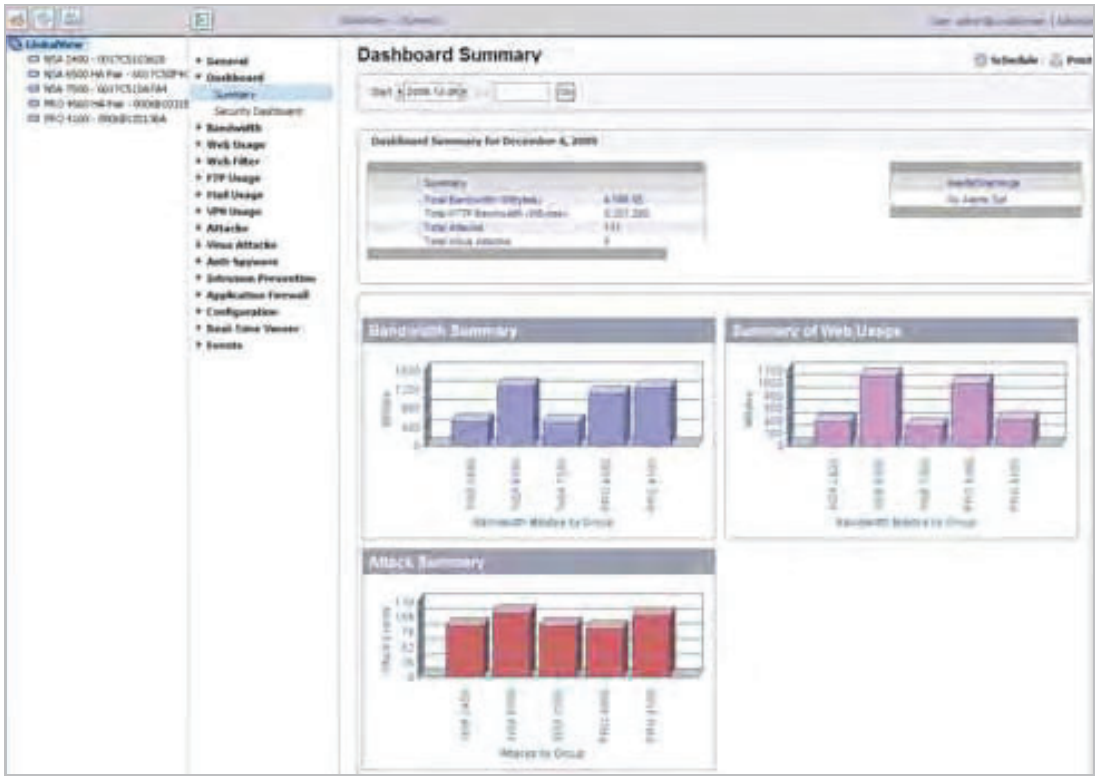



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1364
Keysight, EX-1045
Page 1364

# Global Views

From the Global view of the UTM Panel, Summary and Over Time reports are available for all SonicWALL appliances connected to SonicWALL ViewPoint.

To open the Global view, click the **MyReportsView** icon in the upper-left hand corner of the left pane. .



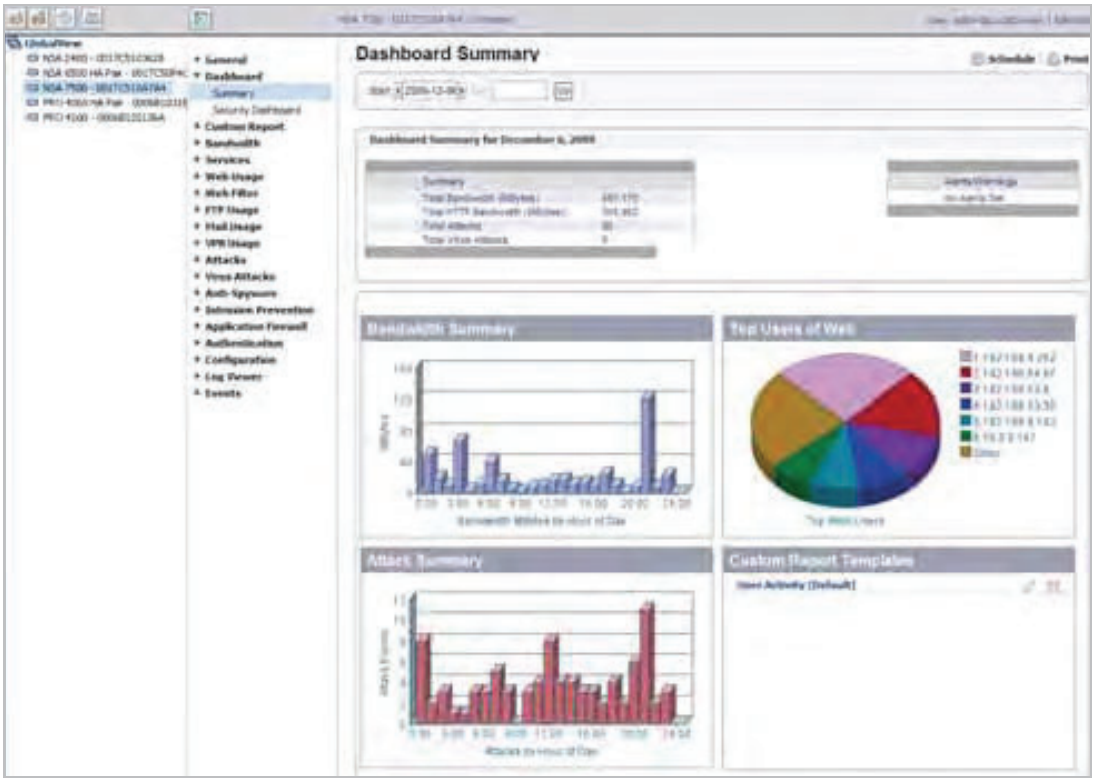

As you navigate the SonicWALL ViewPoint reports screens with the MyReportsView view selected and view different reports, the settings that you specify are maintained in effect throughout the session.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1365
Keysight, EX-1045
Page 1365

# Unit View

From the Unit view of the UTM panel, reports contain detailed data for the selected SonicWALL appliance. To open the Unit view, click the **UTM** tab. Then, click a SonicWALL appliance in the left pane of the SonicWALL ViewPoint interface. The report page for the SonicWALL appliance displays.



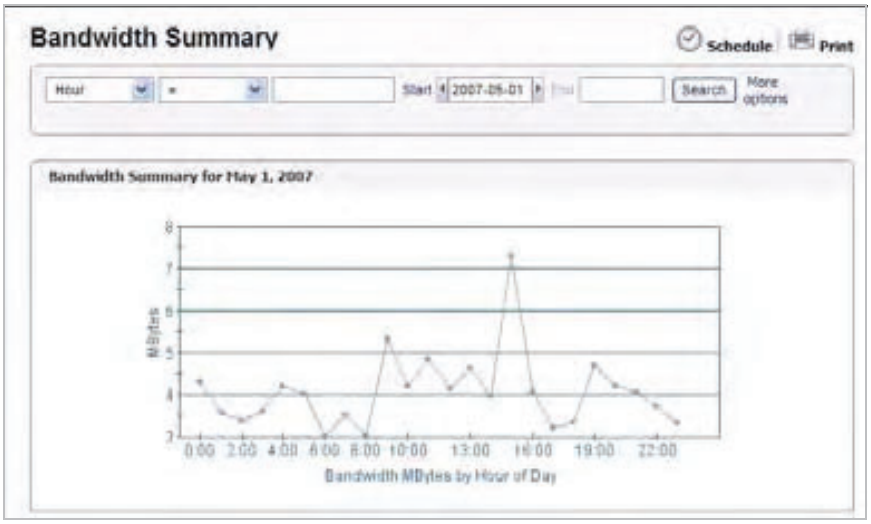

As you navigate the UTM panel with a single SonicWALL appliance selected and change settings, those settings will remain in effect throughout the session.

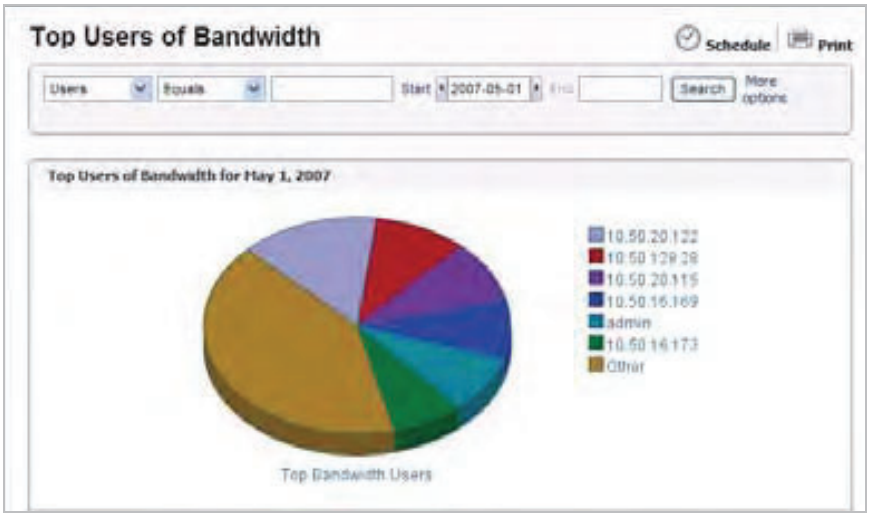


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1366
Keysight, EX-1045
Page 1366

# Using Interactive Reports

ViewPoint provides interactive reporting to create a clear and visually pleasing display of information. The following figures provide examples of an interactive report graph and a pie chart for Summary and Top Users. You can control the way the information is displayed by adjusting the settings which are collapsed in the search bar.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1367
Keysight, EX-1045
Page 1367

# Searching for a Report

The search bar feature provides search and configuration capabilities for every report. In addition to the original quickset functions, the search bar has intuitive search fields to provide context-based searching.



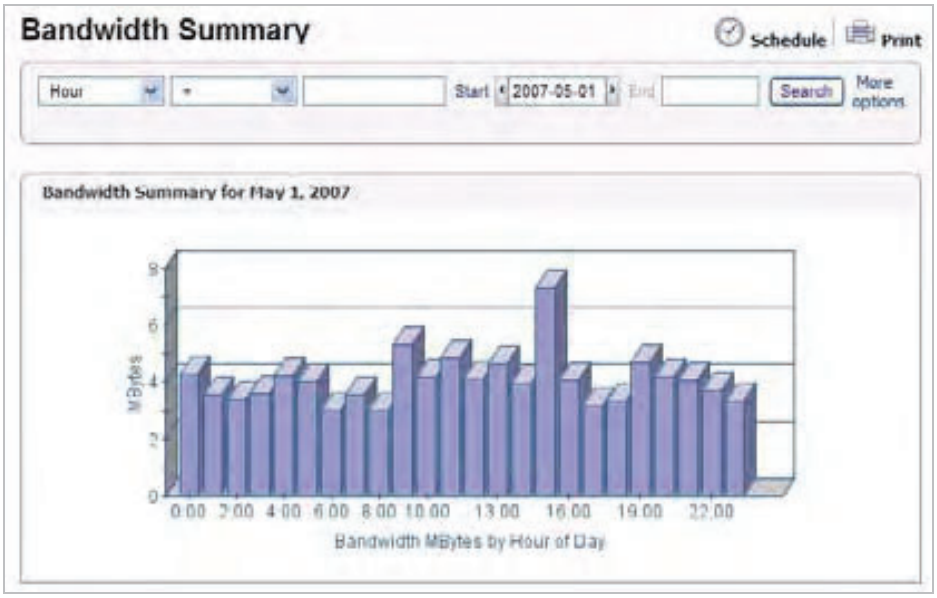

The search bar contains a number of helpful components that allow you to specify search parameters and locate a report with ease. The components of the search bar include:

- A column drop-down list: The searchable column drop-down list contains all the searchable columns of a report. It is context-based, containing different options in different reports. The column drop-down list defines criteria for the search and filter functions.
- An operator drop-down list: There are two types of operator sets. If the content of the selected column is character-based, a character-based list is displayed. If the column contains numerical data, a list with mathematical symbols is displayed.
- A search text field: You can input a search string into this field.
- Start date and end date calendar fields: You can also search for reports by date. Clicking on the **Start** field displays a drop-down calendar where you can select day, month, and year by using the side arrows to navigate. You may also navigate through dates by clicking on the arrows located beside the **start date** and the **end date** fields.
- Detailed drop-down menu



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1368
Keysight, EX-1045
Page 1368

The collapsed and expanded Search Bar views are shown below:





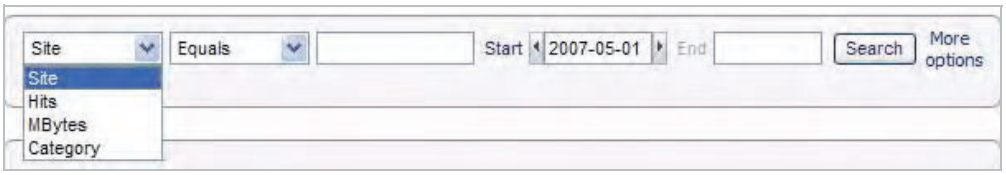




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1369
Keysight, EX-1045
Page 1369

The search bar feature consists of a column drop-down list, an operator drop-down list, a search text field, and a detailed pull-down menu. Search/Filter functions can be performed by utilizing various components reporting at unit level.

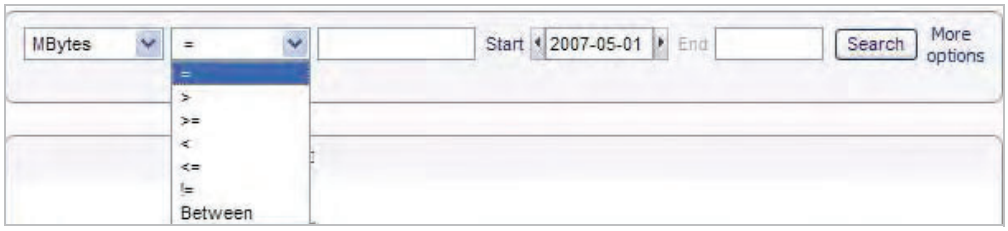
The drop-down list contains all the searchable columns of a report. It is context-based, meaning that it contains different options in different reports. The column drop-down list defines criteria for search and filter functions to work on.




There are two different operator sets. If the content of the selected column is character-based, the character based operators will show:



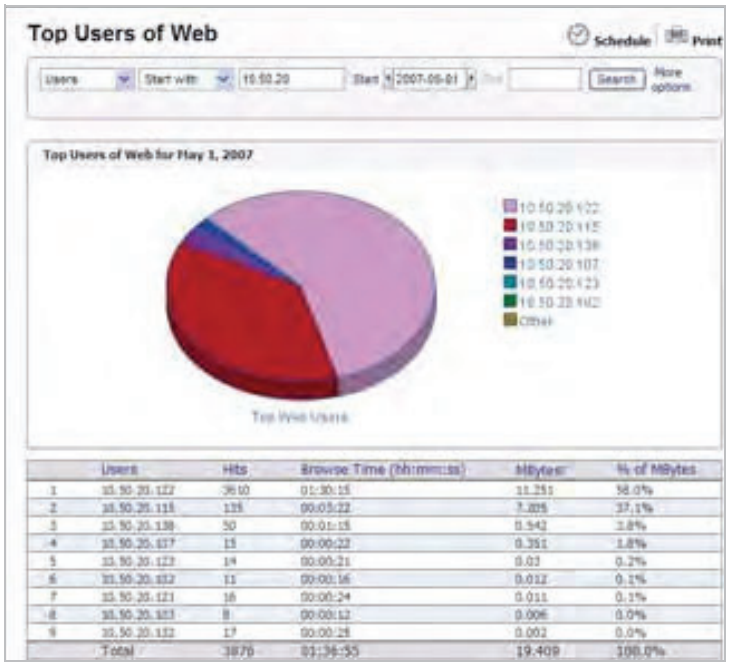

A character-based list contains **Equals**, **Start with**, **End with**, and **Contains** operators. If the content of the selected column contains numerical data, a list with mathematical symbols plus the **between** operator selection will display:



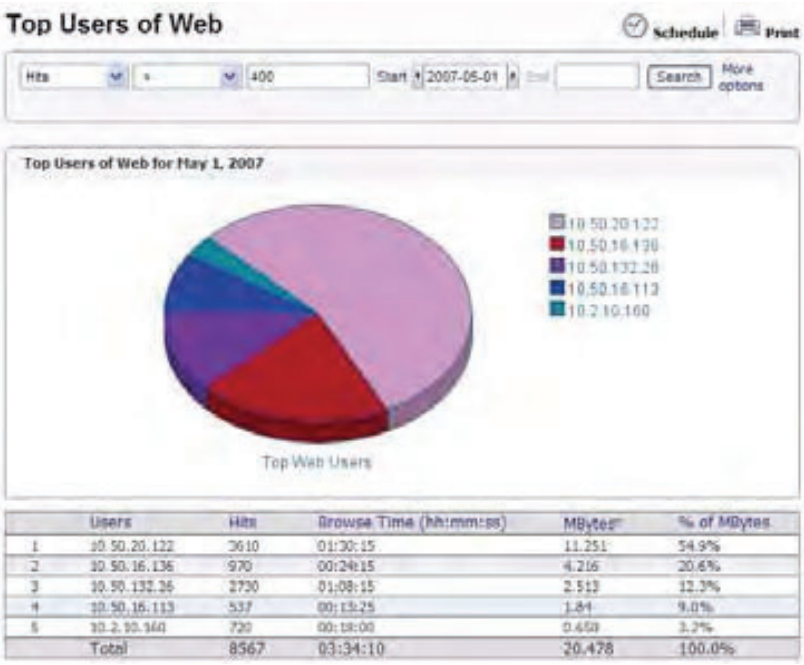



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1370
Keysight, EX-1045
Page 1370

A generated report is shown below with user name (Users) starting with (Start With) “10.50.20” (the value of the search text field).



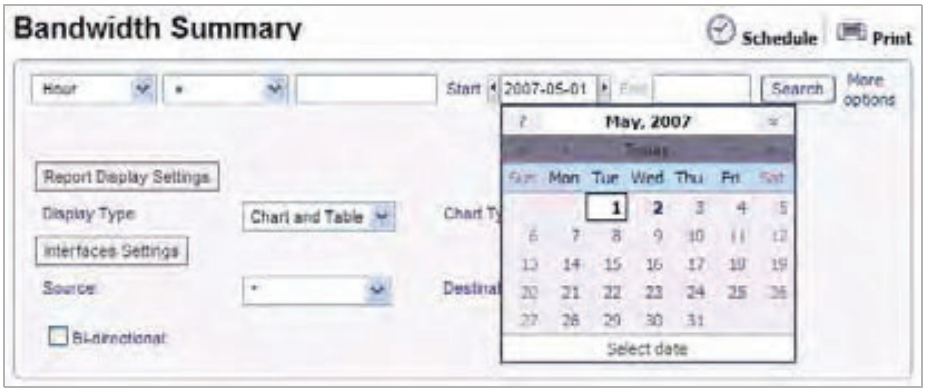

A generated report is shown below in which the Hit count (Hits column) is greater than (>) “100” (the value of the search field).






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1371
Keysight, EX-1045
Page 1371

The calendar module of the search bar is shown below. You can use the calendar module to easily select a date for the Start or End field. You can also manually type in a date. For single day reports, the End field is disabled.




The detailed options are “per report” based. For example, if you select “PIE” as the chart type for report A, you will still see Bar chart in report B if the bar chart was the existing chart type. The detailed drop-down menu can be expanded by clicking **More Options** as shown in the red circle below.

As Figure 1 and Figure 2 show, the options in the detailed drop-down menu are context-based. Figure 1 shows the detailed options of the “Web Usage By User” report. As you can see, Figure 2 contains different options because it is specific to the By User report.

***Figure 1 Context-based Detail Options***






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1372
Keysight, EX-1045
Page 1372

*Figure 2 Web Usage by User - Report Display Settings*




# Collapsible TreeControl Pane

The unit TreeControl pane can be collapsed to free up screen space by clicking on the the small arrow button to the right of the Add Unit, Modify Unit, Refresh, and Find buttons above the TreeControl pane. The panel can be brought back by clicking the same button.

# Enabling/Disabling Scheduled Reports

ViewPoint allows you to disable a scheduled report without deleting it. This allows you to re-use the report at a later time without having to create it again. To enable or disable a report, navigate to the **Configuration > Scheduled Reports** page under the **UTM** tab. This screen shows all the scheduled reports on the current appliance. Select the checkbox in the row for a report(s) that you wish to disable, and click the **Disable Selected Scheduled Reports** button above the table. After confirmation, the check mark in the Enabled column is grayed out. To re-enable the report, use the **Enable Selected Scheduled Reports** button above the table.

# Combined Reports

Users familiar with ViewPoint 4.0 will find two categories of reports that are no longer visible on the function tree: the **Browse Time** report and the **ROI** report. The information from these two reports have been folded into the **Web Usage** and **Bandwidth** reports, respectively. The **Web Usage** report pages now feature a **Browse Time** column. The **Bandwidth** report pages feature a **Cost($)** column that displays all the information previously displayed by the ROI reports.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1373
Keysight, EX-1045
Page 1373

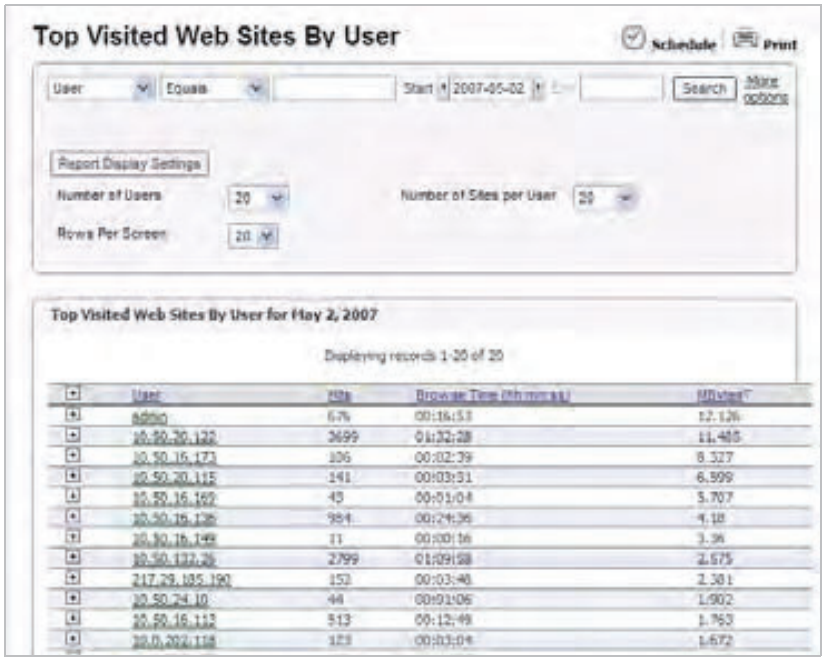
# Improved Navigation

To save time, ViewPoint now features linked reports. **Web Usage** and **Web Filter** reports now link their **By User** and **By Site** pages. It is now possible to navigate directly from the **Web Usage > By User page** to a **Web Usage > By Site** page or from the **Web Filter > By User page** to a **Web Filter > By Site** page detailing the information of the site that the user has been browsing. Click the **Plus** sign next to the entry in the User column to show details, and hover the mouse over a site. A sticky tooltip will display with a link to the corresponding site's report page. This makes navigating from one report to the next much easier and makes retrieving detailed information simple.

## Sample Navigation Use Case

This sample use case demonstrates the improved navigation feature. In this use case you will open up the **Web Usage > By User** report and observe what sites the top browser has been visiting. Then you will move directly from the **By User** report to a detailed **By Site** report.

1. Navigate to the **Web Usage > By User** report from the **UTM** tab.




2. Click the **Plus** button next to any IP address in the User column. This displays detailed information about the sites that the user at that address has been visiting.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1374
Keysight, EX-1045
Page 1374

3. Hover your mouse over a site in this list. Click the **Navigate to Top Visited Web Sites By Site** link to navigate directly to the **Web Usage > By Site** report page.




The **Web Usage > By Site** report page shows detailed information about Web traffic to this site. Information in this report include the IP addresses of users who have browsed that site, as well as how much time they have spent browsing.

# Showing Domain Names in Reports

Reports sometimes show the domain names of systems or websites, and sometimes show only the IP address. This is caused by different firmware versions on the appliances for which reports are being generated.

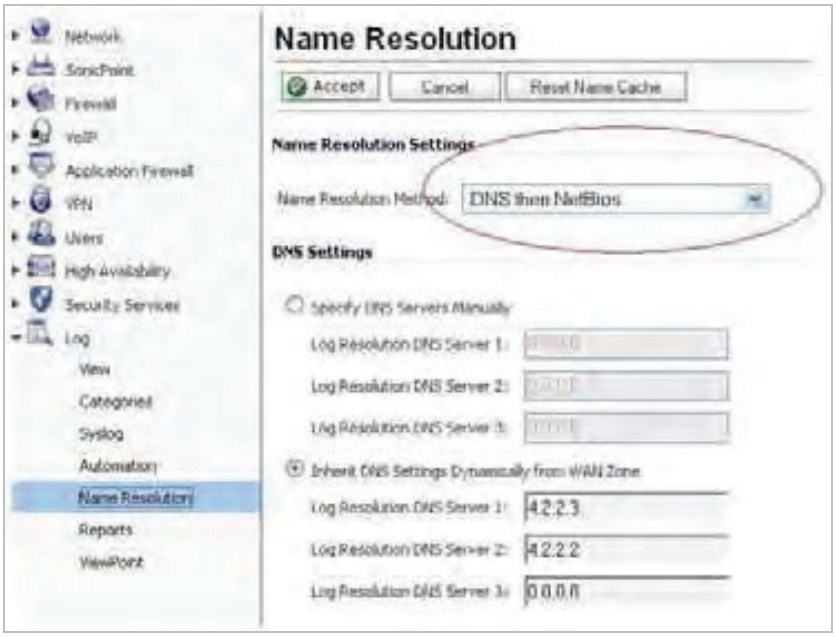
The reporting subsystem consumes the contents of src, dst, dstname, and other tags from the syslog messages. The syslog format and tags depend on the version of the firmware.

For firmware that includes name resolution, the reports will list the domain.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1375
Keysight, EX-1045
Page 1375

**Note** In SonicWALL ViewPoint 5.1 and above, the Name Resolution option on the UTM appliance (where the firmware supports it) is enabled when a unit is added. This does not apply to already existing appliances in the system.




# Managing ViewPoint Reports on the Console Panel

There are management settings for the ViewPoint Reporting Module on the ViewPoint **Console** panel. The UTM panel contains limited configuration screens, used for managing scheduled reports and per-unit settings.The Reports section on the Console panel is divided into sections that allow you to manage system-wide settings, including the following:

***Table 4 Console > Reports***

| Section | Settings |
|---|---|
| **Settings** | Report Settings/Options |
| | Log Viewer Settings |
| **Summarizer** | Summarizer Settings |



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1376
Keysight, EX-1045
Page 1376

| Section | Settings |
|---|---|
| | Reports Data Summarization Interval |
| | Syslog Deletion Schedule |
| | Host Name Resolution Settings |
| **Email/Archive** | Email/Archive Time Settings |
| | Days to Store Archived/Published reports |
| | Email/Archive Configuration - Web Server Details |
| | Logo Settings |
| | SortBy Settings In PDF Reports |
| **Scheduled Reports** | Summary |
| | Search Criteria |
| | Search Results |
| **Management** | Report Data Management Settings |

The Reports section of the Console panel controls settings for syslog data collection, summarizer configuration, email and archiving, scheduling reports, and archiving report data.

- For information about syslog data collection settings, see the "Enabling Report Table Sorting" section on page 72 in the Managing Reports in the Console Panel chapter.
- For information about the summarizer, see the following sections in the Managing Reports in the Console Panel chapter:
  - "About Summary Data in Reports" section on page 73
  - "Summarizer Settings and Summarization Interval" section on page 73
- For information about Email and Archiving settings, see the "Configuring Email/Archive Settings" section on page 81 in the Managing Reports in the Console Panel chapter.
- For a description of how to schedule reports in the Console panel, see the "Scheduled Reports" section on page 82 in the Managing Reports in the Console Panel chapter.
- For information about archiving report data using the Move Data to Archive (MDTA) feature, see the "Management" section on page 87 in the Managing Reports in the Console Panel chapter.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1377
Keysight, EX-1045
Page 1377

# CHAPTER 14
# Scheduling and Configuring Reports

This chapter provides information about scheduling automatic reports and configuring data summarization settings. It also contains instructions for configuring settings for the Dashboard > Summary report and describes how to view the list of current alerts on the **Events > Current Alerts** page. This chapter also describes how to export compliance reports in PDF format. The settings described in this chapter are applied on a per-unit basis.

This chapter includes the following sections:


- "Configuring Scheduled Reports" section on page 134
- "Selecting Reports for Summarization" section on page 137
- "Configuring Inheritance for Reporting Screens" section on page 138
- "Configuring Data Storage Settings" section on page 139
- "Configuring Summarization Data for Top Usage" section on page 140
- "Configuring Summarization Data for Bandwidth Reports" section on page 141
- "Configuring Dashboard Summary Reports" section on page 142
- "Viewing Current Alerts" section on page 144
- "Scheduling PDF Compliance Reports" section on page 144





CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1378
Keysight, EX-1045
Page 1378

# Configuring Scheduled Reports

SonicWALL ViewPoint Reporting can automatically send reports to any email addresses that you specify. This section contains the following:

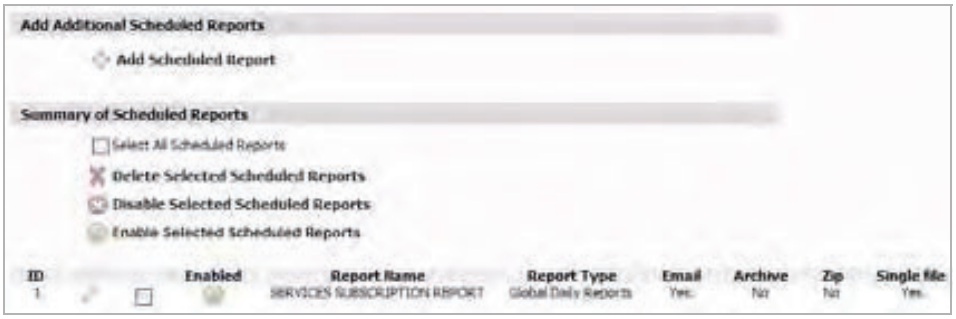
- “Viewing or Managing Scheduled Reports” on page 134
- “Adding or Editing a Scheduled Report” on page 135

To create scheduled email reports in PDF format as *Compliance Reports*, see the “Scheduling PDF Compliance Reports” section on page 144.

## Viewing or Managing Scheduled Reports

To view , delete, or enable/disable currently scheduled reports, perform the following steps:

1. Click the **UTM** tab and select a SonicWALL appliance.
2. Expand the **Configuration** tree and click **Scheduled Reports**. The Scheduled Reports page displays.




3. On the Scheduled Reports page, to add a new scheduled report, click **Add Scheduled Report**. See “Adding or Editing a Scheduled Report” on page 135.
4. To edit a report, click the pencil icon in that row. See “Adding or Editing a Scheduled Report” on page 135.
5. To delete a report, select the checkbox in that row and then click **Delete Selected Scheduled Reports**.
6. To disable a scheduled report, select the checkbox in that row and then click **Disable Selected Scheduled Reports**.
7. To enable a disabled report, select the checkbox in that row and then click **Enable Selected Scheduled Reports**.
8. To select all reports in the list, click **Select All Scheduled Reports**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1379
Keysight, EX-1045
Page 1379

# Adding or Editing a Scheduled Report

You can add a new scheduled report or edit an existing one on the UTM panel on the Configuration > Scheduled Reports screen. When adding or editing the report, you can configure its name, category, formats, cover page, summary report page, and detailed reports page. You can also use or create a profile for the detailed reports page settings.

To add or edit a new scheduled report, perform the following steps:

1. Navigate to the Configuration > Scheduled Reports page and do one of the following:
   - To add a new schedule report, click the **Add Scheduled Report** button.
   - To edit an existing report, click the pencil icon in that row. The Scheduled Report Configuration window displays.
2. Enter a name for the report in the **Name** field.
3. Enter descriptive information in the **Description** field.
4. To email the report, select the **Email** check box. The screen expands to show email configuration settings.
5. Enter the IP address of the mail server into the **SMTP Server** field.
6. By default, the ViewPoint Reporting Module will use the email address that was configured in the Console panel in the Management > ViewPoint Settings screen as the Sender email address. To change it, enter a new Sender email address in the **Source Email Address** field.
7. Enter one or more destination email addresses, separated by semicolons, into the **Destination Email Addresses** field.
8. Enter the Subject Line that will appear in reports sent from the ViewPoint Reporting Module in the **Email Subject** field.
9. Enter text that will appear in the message body in the **Email Body** field.
10. To copy the contents of the report into the body of the email message, select the **Send Reports Inline** check box. To send the file as an email attachment, make sure this check box is deselected.



**Note** Reports can only be sent inline when all data is sent in a single report.

11. To archive the file on the server's hard disk, select the **Archive** check box and enter a path in the **Save Directory** field.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1380
Keysight, EX-1045
Page 1380

Specify the directory where the file will be archived in the **Save Directory** field.

12. For Report Type, select **Daily**, **Weekly**, or **Monthly**.
13. For Report Format, select **HTML**, **XML**, or **PDF**.
14. Select either **Include all data in a single report** or **Zip Reports into a single file**.
15. If you selected **PDF** for the Report Format, you can create a password to protect it by selecting **Password Protect the PDF File** and typing a password into the **Password** field. Users must input the password to view the contents of a password-protected PDF file. The content can be copied or printed, but is not editable by a PDF editor.
16. If the zip file is selected, you can create a password for it by selecting **Password Protect the Zip File** and typing a password into the **Password** field.



**Note** When both **PDF** and **Zip Reports into a single file** are selected, you can password-protect the PDF, but not the zip file.

17. For the Cover Page, enter a **Title** and **Subtitle** and select colors for the **Foreground** and **Background** of the cover page.
18. For Summary Report Page, you can select up to 4 reports. Select a report for the summary page from the **Choose the Summary Reports** drop down list, and then click **Add**.
19. For Detailed Report Page, do one of the following:
    - Click **Select an existing profile**, and then select the profile to use from the **Profile Name** drop-down list.
    - Click **Create a new profile**, type a profile name into the **New Profile Name** field, and then select the checkboxes in the Report list for each report to be included. You can click the checkbox next to the **Report** heading to select all reports in the list.
20. Optionally click **Configure Filters Options**. For this procedure see “Configuring Filters and Options” on page 137.
21. To see a preview of this scheduled report, click **PREVIEW**.
22. When finished, click **Add**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1381
Keysight, EX-1045
Page 1381

## Configuring Filters and Options

1. At the bottom of the Scheduled Report Configuration page, click the **Configure Filters/Options** button. The Display Options/Settings page displays.
2. Select the number of sites to display in Top Sites reports (default: 20).
3. Select the number of users to display in Top Users reports (default: 20).
4. Select the number of sites to display in Sites by User/Users By Site reports (default: 20).
5. Select the number of items to display in all other reports (default: 20).
6. Select the number of entries per item to display in all other reports (default: 20).
7. Under Inclusion Filter Parameters, enter a comma separated list of sites to include in By Site reports in the **Site List** field.
8. Enter a comma separated list of users to include in By User reports in the **User List** field.
9. To include the user's full name and IP address in the report, select the **Whole Name/IP** checkbox.
10. For Bandwidth Usage reports, select the source from the **Source Interface** drop-down list.
11. For Bandwidth Usage reports, select the destination from the **Destination Interface** drop-down list.
12. Click the **Update** button to apply changes. The new report will appear in the list on the Scheduled Reports page.

# Selecting Reports for Summarization

This section describes how to tune the performance of the Summarizer by configuring which reports will be created. When an appliance is configured to communicate with ViewPoint, you need to prepare it for syslog data collection for reporting. Make sure the summarizer is collecting data for the reports you want for this unit.

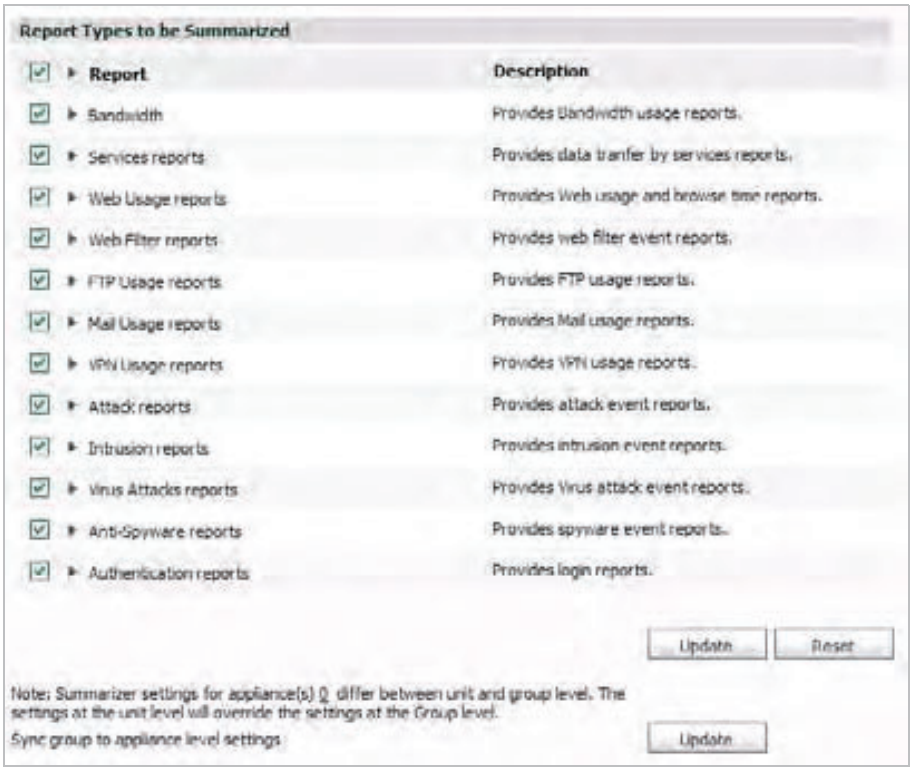
To configure the Summarizer settings, perform the following steps:

1. Click the **UTM** tab.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1382
Keysight, EX-1045
Page 1382

2. Expand the **Configuration** tree and click **Summarizer Settings**. The Summarizer Settings page provides a list of reports and a correlating description of each report. Each report contains a checkbox that you can select to generate a summarized report.




3. Select the checkbox of each report type to summarize.
4. When you are finished, click **Update**. Your configuration changes are saved automatically.

# Configuring Inheritance for Reporting Screens

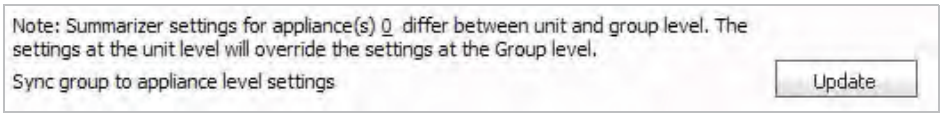
On the Configuration > Summarizer Settings screen, there is an option to synchronize report settings between the unit level and global level. This option can be displayed in any of the sections on this page when those settings are not synchronized between the unit level and global level.

This option provides inheritance support for report settings.

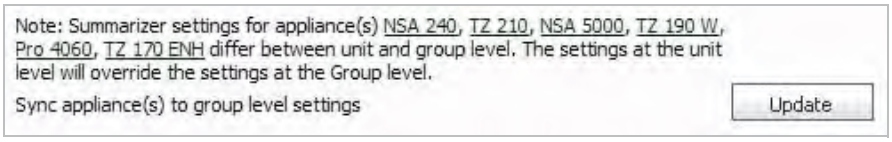


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1383
Keysight, EX-1045
Page 1383

When you are viewing the screen at the unit level, the option is **Sync group to appliance level settings**. This is reverse inheritance. Click the **Update** button to apply your current unit level settings to the group to which this unit belongs.




When you are viewing the screen at the global level, the option is **Sync appliance(s) to group level settings**. This is forward inheritance. Click the **Update** button to apply your current global level settings to the appliances in this group.




# Configuring Data Storage Settings

The Data Storage Configuration section of the **Configuration > Summarizer Settings** page allows you to specify the number of days to store summarized data and syslog data.




For all fields in this section, the minimum values should be 3 days, and will typically be longer.

Raw syslog data is transferred to the ViewPoint system by individual SonicWALL appliances, where it is stored in raw syslog files. The data from these files is combined and stored in a raw syslog database. Data from this database is processed by the Summarizer and then stored in the summarized data database.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1384
Keysight, EX-1045
Page 1384

The raw syslog files and databases older than the number of days specified here will get deleted by the global daily deletion schedule configured on the Console > Reports > Summarizer page. That page also provides a way to delete the summarized database for a certain date. See the "Configuring the Syslog Deletion Schedule Settings" section on page 78.

To configure the Data Storage Configuration settings:

1. On the **UTM** tab, expand the **Configuration** tree and click **Summarizer Settings**.
2. Scroll down to the **Data Storage Configuration** section.
3. Type the desired number of days to store summarized data into the **Days To Store Summarized Data** field and then click **Update**.
4. Type the desired number of days to store raw syslog database files into the **Days To Store Raw Syslog Databases** field and then click **Update**.
5. Type the desired number of days to store raw syslog database files into the **Days To Store Raw Syslog Databases** field and then click **Update**.
6. Type the desired number of days to store archived XML reports into the **Days To Store XML reports** field and then click **Update**.

# Configuring Summarization Data for Top Usage

The Reports Summarization Data for Top Usage section of the Configuration > Summarizer Settings page allows you to enable Web event consolidation and resolve unrated categories.

When enabled, Web event consolidation reduces repetitive syslog event entries within the syslog database. Enabling Web Event Consolidation promotes search and summarizer efficiency by consolidating the syslog messages that result from a single click (for example, a visit to a Web page), and further correlates events by time proximity, such as multiple visits to the same URL by the same user within a set time, and HTTP header information. ViewPoint consolidates syslog messages under the main domain name.

When Web Event Consolidation is disabled, multiple syslog events are logged for one request. For instance, a single access to www.cnn.com can generate more than 70 syslog messages. Many of the 70 syslog messages refer to the links to other pages like images.cnn.com or video.cnn.com that are included in the Web page. In this simplified example, if **Domain Only** consolidation is



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1385
Keysight, EX-1045
Page 1385

selected, then only one Web event is recorded (cnn.com). If **Host & Domain** is selected, then you would see three Web events. You would see all 70 Web events if consolidation was not enabled at all.

To enable Web event consolidation and resolve unrated categories, perform the following:

1. On the **UTM** tab, expand the **Configuration** tree and click **Summarizer Settings**.
2. Scroll down to the **Reports Summarization Data for Top Usage** section.
3. Select the **Enable Web Event Consolidation** checkbox to consolidate repetitive syslog event entries within the syslog database and then select one of the following levels of consolidation:
   - Host & Domain - More restrictive, less consolidation
   - Domain Only - More general, more consolidation
4. Optionally select the **Resolve "Not Rated" categories using message comparison** checkbox. If enabled, ViewPoint will attempt to categorize unrated items by comparing them to rated items, and will display the results in reports.
5. Click **Update**.

# Configuring Summarization Data for Bandwidth Reports

The Reports Summarization Data for Bandwidth Reports section of the Configuration > Summarizer Settings page allows you to configure the currency type and cost per megabyte for use in bandwidth reports.

To configure the data for bandwidth reports, perform the following:

1. On the **UTM** tab, expand the **Configuration** tree and click **Summarizer Settings**.
2. In the **Reports Summarization Data for Bandwidth Reports** section, select the currency type in the **Type of Currency** field. Over 20 different currencies from around the world are available.
3. Specify an amount based on your chosen currency in the **Cost Per Mega Byte Bandwidth Use** field.
4. Click **Update**.

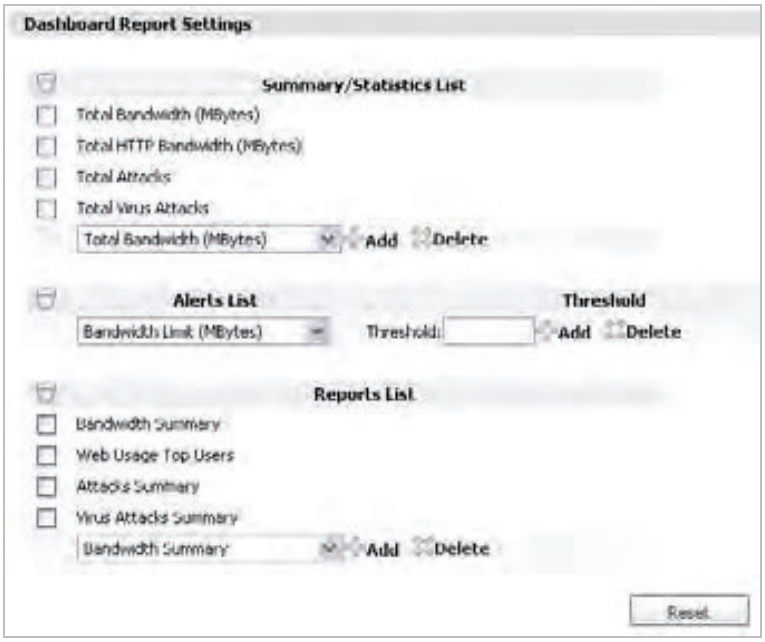


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1386
Keysight, EX-1045
Page 1386

# Configuring Dashboard Summary Reports

In the Configuration > Dashboard page, you can configure settings to control the information displayed by the Dashboard > Summary screen. Settings are available for the following:

- Summary statistics list at the top left of the Dashboard > Summary page
- Alerts list at the top right of the Dashboard > Summary page
- Reports list in the main body of the Dashboard > Summary page



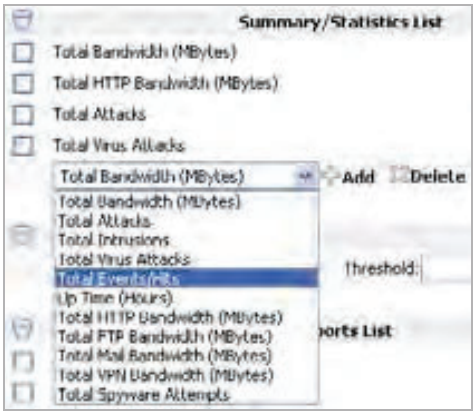

To configure Dashboard Summary report settings, perform the following steps:

1. Click the **UTM** tab.
2. Expand the **Configuration** tree and click **Dashboard**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1387
Keysight, EX-1045
Page 1387

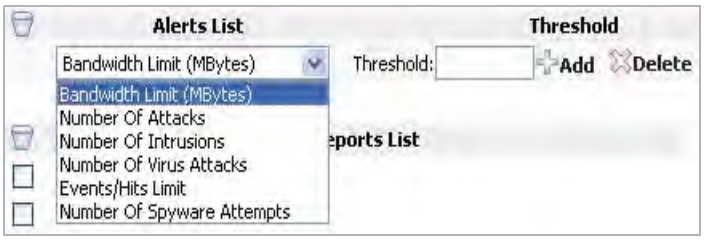
3. In the Summary / Statistics List section, to add a statistic to the Dashboard > Summary page, select it from the drop-down list and then click **Add**.




4. To remove a statistic from the Dashboard > Summary page, select the checkbox under the trashcan icon for that statistic, and then click **Delete**.

5. In the Alerts List section, to add an alert to the Dashboard > Summary page and to receive an email alert when the alert setting is matched, select an event type from the drop-down list, type a threshold value into the **Threshold** field, and then click **Add**.

   Alerts are emailed using the settings configured in the Console > Management screens. See “Settings” on page 61 and “Alert Settings” on page 64.




6. To remove an alert, select the checkbox under the trashcan icon for that alert, and then click **Delete**.

7. In the Reports List section, to add a report to the Dashboard > Summary page, select the report type from the drop-down list, and then click **Add**.

8. To remove a report from the Dashboard > Summary page, select the checkbox under the trashcan icon for that report, and then click **Delete**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1388
Keysight, EX-1045
Page 1388

# Viewing Current Alerts

You can view a list of current alerts on the **Events > Current Alerts** page of the UTM panel. Select a global view or unit to view current alerts for your selection.




# Scheduling PDF Compliance Reports

ViewPoint can create scheduled email reports in PDF format. Called *Compliance Reports*, this feature allows you to export regular reports in universally readable format.

## Compliance Report Overview

A *Compliance Report* is a report that collects report data and presents it in an organized format.

The ViewPoint Compliance Report feature allows administrators to provide more customized report summaries and to create more formal and defined layout of report information in PDF format. This feature provides the following benefits:

- Customizable cover page (Default also available)
- Customize Summary/ Descriptions for the reports.
- Ability to customize a set of reports.
- Three reports can be persisted as a profile so that it can be consumed by less experienced users in the system.
- Reports can be generated in industry standard PDF format.
- Compressed format provides a smaller sized file than an equivalent HTML report.
- The print quality is higher.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1389
Keysight, EX-1045
Page 1389

- This feature has the ability to open a 200 page PDF report with ease. In comparison, opening the same report in HTML takes a more extensive amount of time using IE, as it is weighed down by memory and other systems.

## Requirements

Adobe Reader ® plug-in is required for the preview function.

## How Do Compliance Reports Work?

ViewPoint has the capability to generate both online and scheduled reports in HTML format. Since PDF has become a standard document format for distribution, the compliance reports are based on this universal standard. Moreover, users are able to customize/define sections throughout the report. For example, they can assign different logos/titles to the cover pages for their customers.

# Adding a New Scheduled Compliance Report

This section includes the following sub-sections:


- "Customizing Your Cover Page" section on page 147
- "Customizing Your Summary Report Page" section on page 148
- "Customizing Your Detailed Reports Page" section on page 149
- "Editing Existing Profiles" section on page 150
- "Verifying User Compliance Reports Configuration" section on page 152

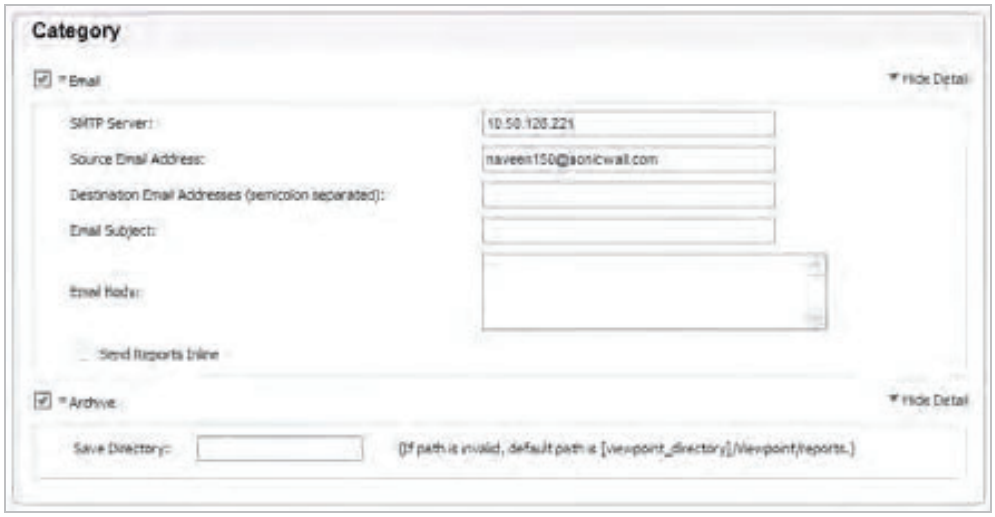

To begin creating a new customized Compliance Report, perform the following steps:

1. Navigate to **UTM > Configuration > Scheduled Reports.**
2. Click the **ADD** button, to add a scheduled report.
3. The Scheduled Report Configuration page displays. In the **General** section, enter the name of your report into the **Name** field, and the report description.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1390
Keysight, EX-1045
Page 1390

4. In the **Category** section, select the **Email** check box. The details window displays:
   - **SMTP Server** field: Enter your SMTP Server IP address or hostname.
   - **Source Email Address** field: Enter your Source Email Address.
   - **Destination Email Address** field: Enter the Destination Email Address(es).
   - **Email Subject** field: Enter your Email Subject.
   - **Email Body** field: Enter your Email Body.




5. To archive a directory, click the **Archive** check box. Enter the your desired directory you want to archive into the **Save Directory** field.

To change the format and settings of your customized compliance report, perform the following steps:

6. In the **Format and Settings** category, select the **Report Type** that reflects the time interval you want to view your reports, either **Daily**, **Weekly**, or **Monthly**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1391
Keysight, EX-1045
Page 1391

7. Select the PDF report format in the **Report Format** category. Selecting the **PDF** option will open additional fields to allow you to customize the set up of the **Cover Page**, **Summary Report Page**, and **Detailed Report Page** of your report in PDF format.




8. To zip all of your reports into a single file, select the check box next to the **Zip Reports into a single file** check box.

**Note** PDF will disable some options that are only applicable to HTML.

9. For custom reports, enter the template folder name into the **Template Folder Name** field.

## Customizing Your Cover Page

The Cover Page section allows the user to design a cover page for their report using different color schemes.

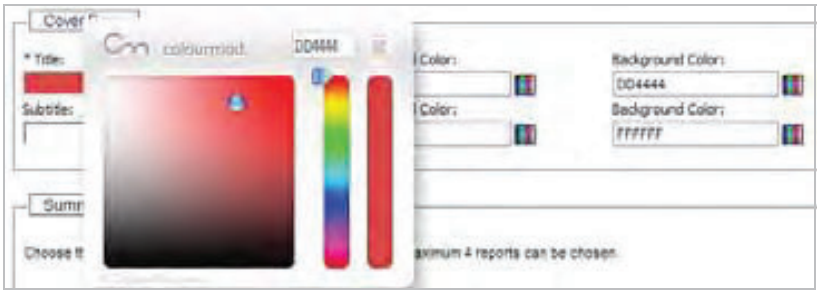
1. **Title** field: Enter the document title.
2. **Subtitle** field: Enter the document subtitle. (Optional).






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1392
Keysight, EX-1045
Page 1392

3. Select the color for the Title and Subtitle's foreground and background by clicking the gradient color box in the right side of the each field. You may select a color by either choosing a color on the color bar and then selecting its value in the color box or by typing in the HTML color.

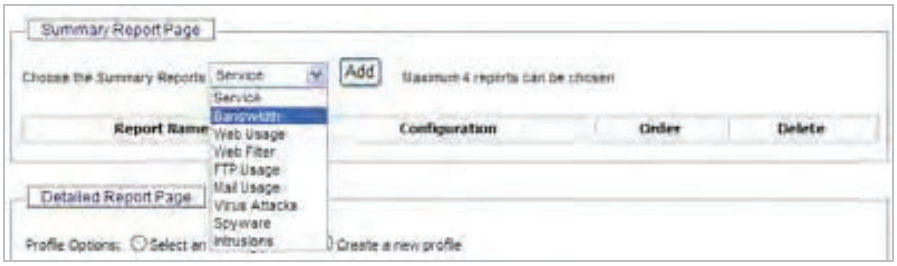



4. The color codes are automatically filled in the corresponding fields once the color chooser window is closed.

## Customizing Your Summary Report Page

The Summary Report Page allows you to add new reports and individually customize their appearance.

1. On the Summary report page, select the type of summary reports you need, up to a maximum of 4 reports. Then, click the **Add** button. The report will be created based on the type of summary report you have selected.
2. Enter the report title in and report description in the appropriate fields.
3. Select the text color for the title and description.
4. Select the background color for both fields.
5. Select the order in the **Order** drop-down window.




6. You may continue to add reports based on the summary you select in the Summary Reports drop-down menu. Repeat steps 1-5 to add more summary reports.

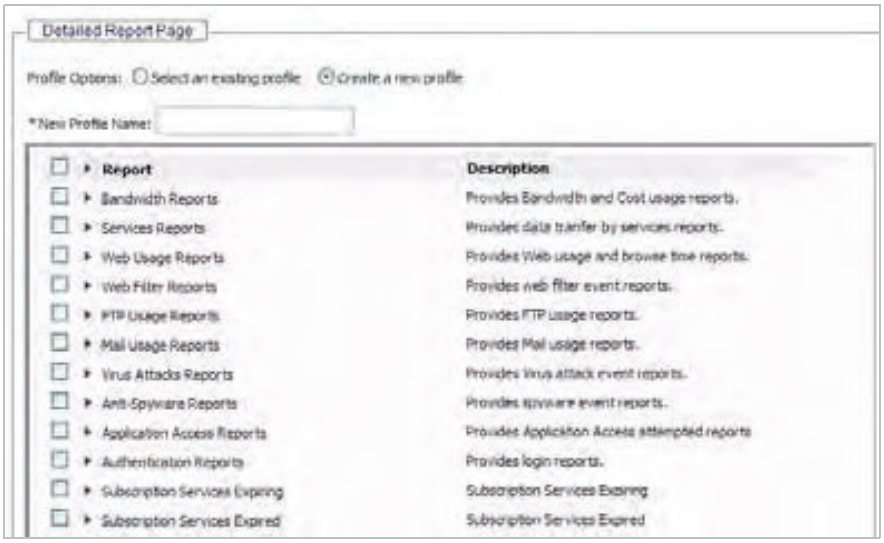


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1393
Keysight, EX-1045
Page 1393

# Customizing Your Detailed Reports Page

The Details Report Page provides you with a list of reports you may select to include in your report summaries. You can refine your setting for your report in more detail in the Detailed Report Settings category. First, select the appropriate profile setting for your report. If you are creating a new profile, select the **Create a New Profile** button.

1. **New Profile Name** field: Enter the name of your new profile.



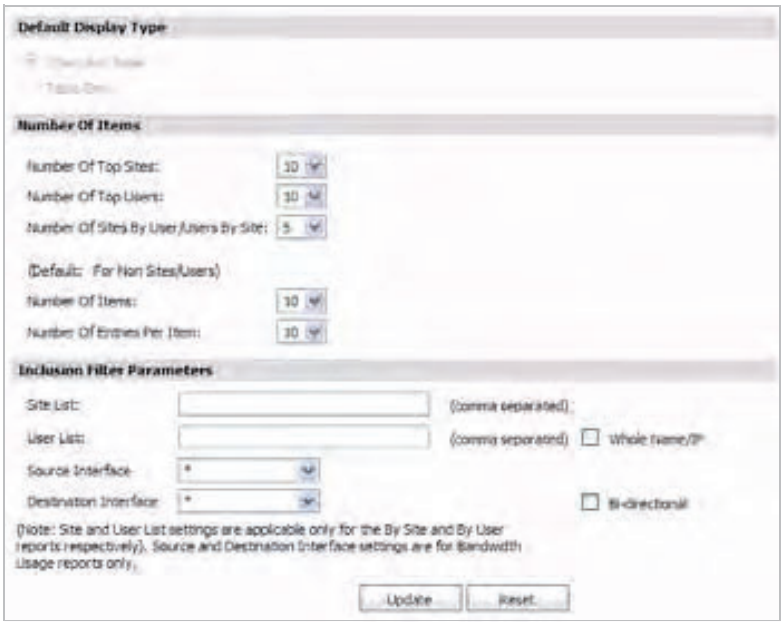

2. To determine the type of reports that will be summarized in your compliance report, check the boxes next to the reports you need. Sub-folders are revealed to each folder by clicking the plus icon. When all sub-folders are selected, the main folder will be selected.
3. When you have completed your selection(s) of reports, scroll down the page until you see a check button with Configure Filters/Options beside it. Click the check mark button.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1394
Keysight, EX-1045
Page 1394

4. In the Configure Filter/Options section, you are able to decide how your filter and display is set. Once you have clicked the check button, fill out the table accordingly.




## Editing Existing Profiles

A profile is associated with selected reports from the report list. You have the ability to go back and edit existing profiles in your scheduled reports. Since the report list is populated based on the report type selection, a profile is associated with the report type also. Instead of three categories, there will only be two: **single day** or **multi-days**. A profile in a single report will not be seen be seen by the users when they select weekly or monthly as report types.

To edit existing profiles, perform the following tasks:

1. Click the **Edit** icon, located next to the report name you want to edit.



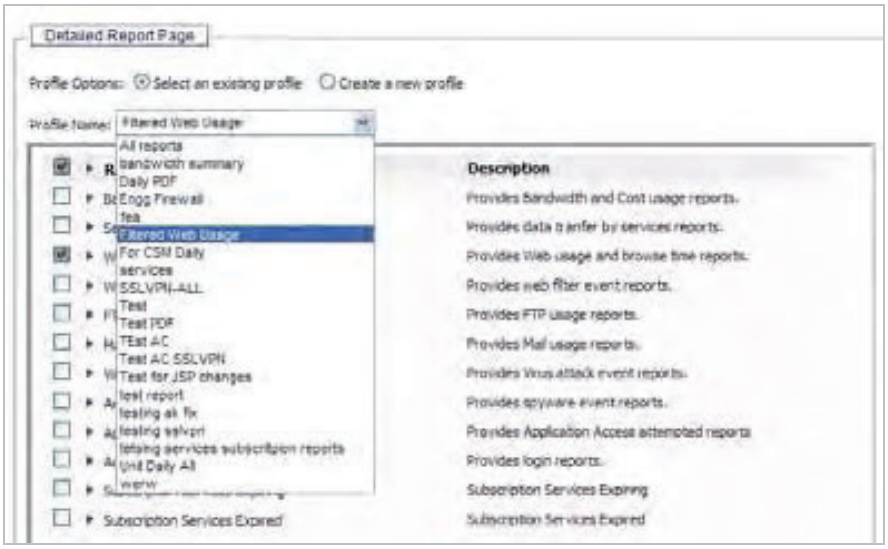

2. In the Detailed Page section, choose the **Select an existing profile** button.

**Note** You are able to delete an existing profile in that section by clicking the **Delete Selected Scheduled Reports** button located at the top of the page.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1395
Keysight, EX-1045
Page 1395

3. From the drop-down list in the Detailed Report Page, select the profile name you wish to edit. Choose the reports you want to add or remove from that profile. If a new profile has the same name as one of the existing profiles, the behavior will be the same as users opening the existing profile and edit the report list. When selecting an existing profile, the associated reports are checked in the report list automatically.




A default cover page is provided:






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1396
Keysight, EX-1045
Page 1396

# Verifying User Compliance Reports Configuration

If you have chosen the PDF version of this report, you now have the option to see a preview of the report covers you have created and how all of the report summaries you added will fit into that template.

To review your customize PDF settings, click the **Preview** button:

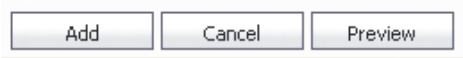



***Figure 3 Cover page; Summary page; and Details page Preview***




**Note** The images used for the preview do not use actual data.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1397
Keysight, EX-1045
Page 1397

# CHAPTER 15
# Viewing Reports

This chapter describes how to generate reports using the SonicWALL ViewPoint Reporting Module.

The following section describes how to configure the settings for viewing reports:


- "Managing Report Settings" section on page 154


Select from the following reports:


- "Viewing General Status Reports" section on page 157
- "Viewing Dashboard Reports" section on page 159
- "Using Custom Reports on UTM Appliances" section on page 163
- "Viewing Bandwidth Reports" section on page 180
- "Viewing Services Reports" section on page 189
- "Viewing Web Usage Reports" section on page 191
- "Viewing Web Filter Reports" section on page 209
- "Viewing File Transfer Protocol Reports" section on page 225
- "Viewing Mail Usage Reports" section on page 231
- "Viewing VPN Usage Reports" section on page 238
- "Viewing Attacks Reports" section on page 250
- "Viewing Virus Attacks Reports" section on page 260
- "Viewing Anti-Spyware Reports" section on page 266





CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1398
Keysight, EX-1045
Page 1398

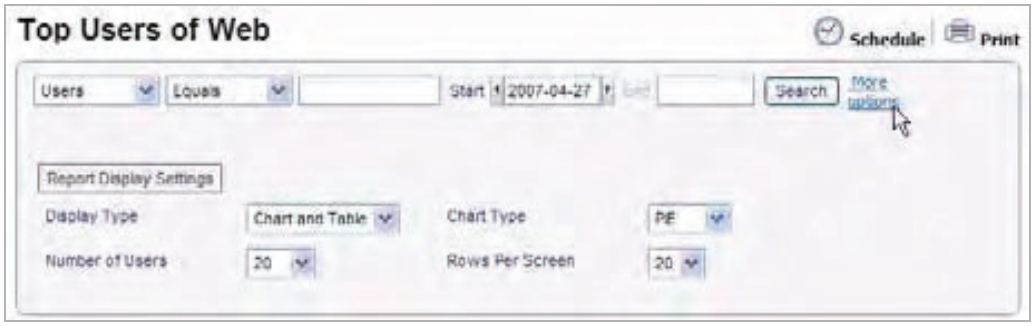
- “Viewing Intrusion Prevention Reports” section on page 273
- “Viewing Application Firewall Reports” section on page 281
- “Viewing Authentication Reports” section on page 287
- “Viewing the Log” section on page 290


# Managing Report Settings

All of the reports in ViewPoint report on data gathered on a specific date or range of dates. You can also edit the report settings for each report by using the Search Bar and the More Options button.

## Editing Report Settings

To edit the report settings, use the Search Bar at the top of the report. You can search other reports, set the start and end dates for a report to view, or click **More Options** to access other Report Display Settings. For a detailed description, see the “Searching for a Report” section on page 123.



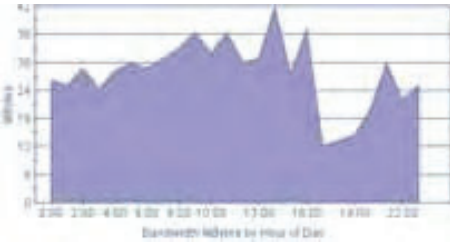

## Selecting a Graphical Display

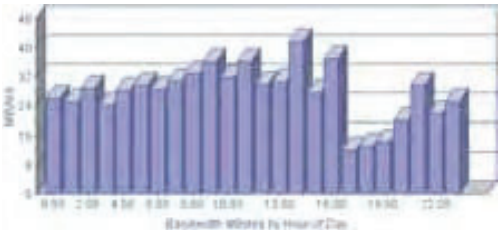
Some reports allow you to specify how many items to display in the report. Select **5**, **10**, **20**, **50**, **100**, or **All** from the **Number of Items** list. This allows you to limit the display to a the specified number in order to make the report easier to read.



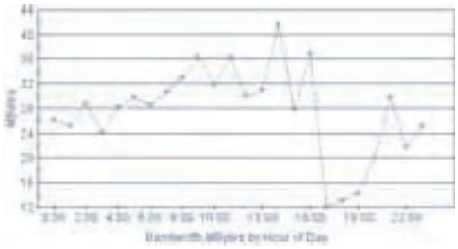
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1399
Keysight, EX-1045
Page 1399

Many reports offer different graphical displays for the data, such as a bar-graph or a pie chart. To select a graphical display, select **Chart and Table** under **Report Display Settings** and choose the display type from the **Chart Type** list. Your selection should display immediately in the report screen. For most reports you can choose **Area**, **Bar**, **Pie** or **Plot**.







# Setting a Date or Date Range

Summary reports display only information for a single date. Over-time reports display information over a date range.

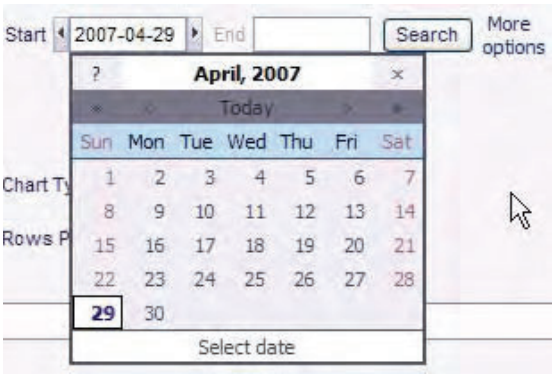
## Selecting a Single Date

To select a single date for a report, click on the **Start** or **End** fields in the Search Bar to display the drop-down calendar. The End field is only configurable for Over Time reports. In the calendar, you can set the month by



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1400
Keysight, EX-1045
Page 1400

clicking the single arrows (<, >), or the year by clicking the double arrows (<<, >>). To select the month or year from a drop-down list, click and hold the arrow button. Click **Search** to begin building the report.




## Selecting a Date Range

To select a date range for an Over Time report, select a **Start Date** and **End Date** in the Search Bar, and then click **Search**. You can use the drop-down calendars by clicking in either field.

# Additional Settings

Many reports have additional settings that you can select such as source and destination interfaces to report traffic through or how to display names and IP addresses. Make your selection from these lists and click **Search**.

# Troubleshooting Reports

One of the most common error messages when a report does not display is “No Data”. There are several reasons why you might see this error, and SonicWALL ViewPoint 5.1 and higher displays the most likely reason and points you to the screen where you can make the necessary adjustments.

Some examples are shown in the following figures.

***Figure 4 Appliance is Not Licensed for Reporting***






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1401
Keysight, EX-1045
Page 1401

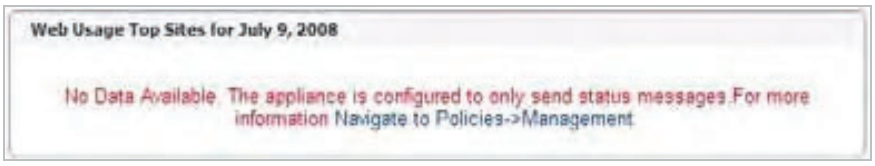
*Figure 5 Appliance is Down*




*Figure 6 Appliance in a Provisioned State*




*Figure 7 Configured for Status Only*




# Viewing General Status Reports

The **General > Status** page contains information on the SonicWALL appliance or group of SonicWALL appliances.

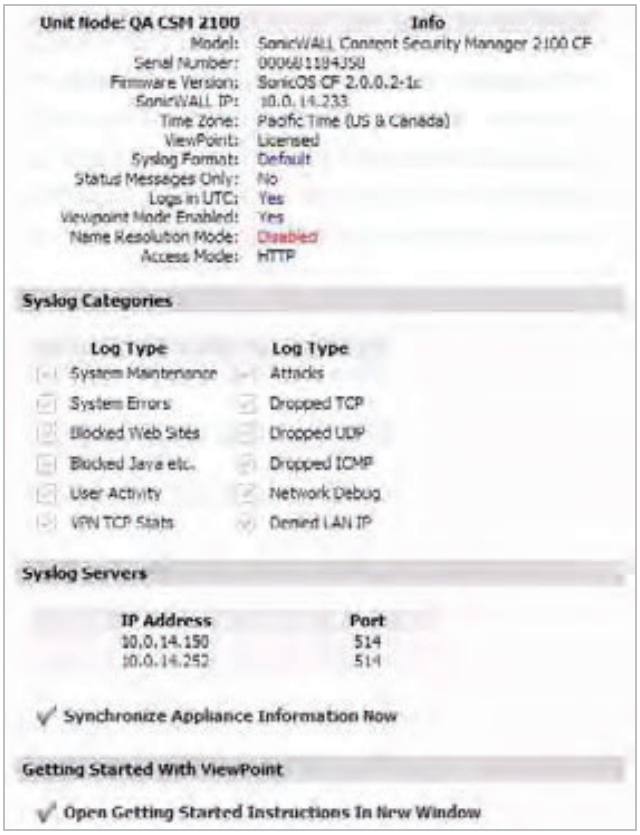
To view the Status page, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1402
Keysight, EX-1045
Page 1402

3. Expand the **General** tree and click **Status**. The Status page displays.




4. The sections contain the following information:
   - Node information—Information on the firewall(s) is displayed at the global or unit level.
   - Syslog Categories—The types of syslog data selected to be collected for the selected appliance.
   - Syslog Servers—The IP address and Port number of the syslog servers configured to collect data from the selected appliance.
     - Synchronize Applicance Information with ViewPoint—Click the **Synchronize Applicance Information Now** link to refresh status data about the monitored appliances. This status information is normally updated every 24 hours.
   - Getting Started With ViewPoint—Click the **Open Getting Started Instructions In New Window** link to open the ViewPoint installation and initial configuration instructions in a separate window.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1403
Keysight, EX-1045
Page 1403

# Viewing Dashboard Reports

Dashboard reports display an overview of bandwidth, uptime, intrusions and attacks, and alerts for managed SonicWALL UTM appliances. The Security Dashboard report provides data about worldwide security threats that can affect your network. The Dashboard also displays data about threats blocked by the SonicWALL security appliance.

Select from the following:


- "Viewing the Dashboard Summary Report" on page 159
- "Viewing the Security Dashboard Report" on page 162


## Viewing the Dashboard Summary Report

The Dashboard Summary report displays statistics, alerts, graphical summary reports, and a list of available custom report templates. Displayed statistics can include total bandwidth, total attacks and other measurable information. The alerts list is displayed when the configured threshold has been reached. A wide range of graphical reports are also available for display.

You can configure the Dashboard > Summary report contents in the UTM > Configuration > Dashboard page. For a description of the configuration procedure, see "Configuring Dashboard Summary Reports" section on page 142.

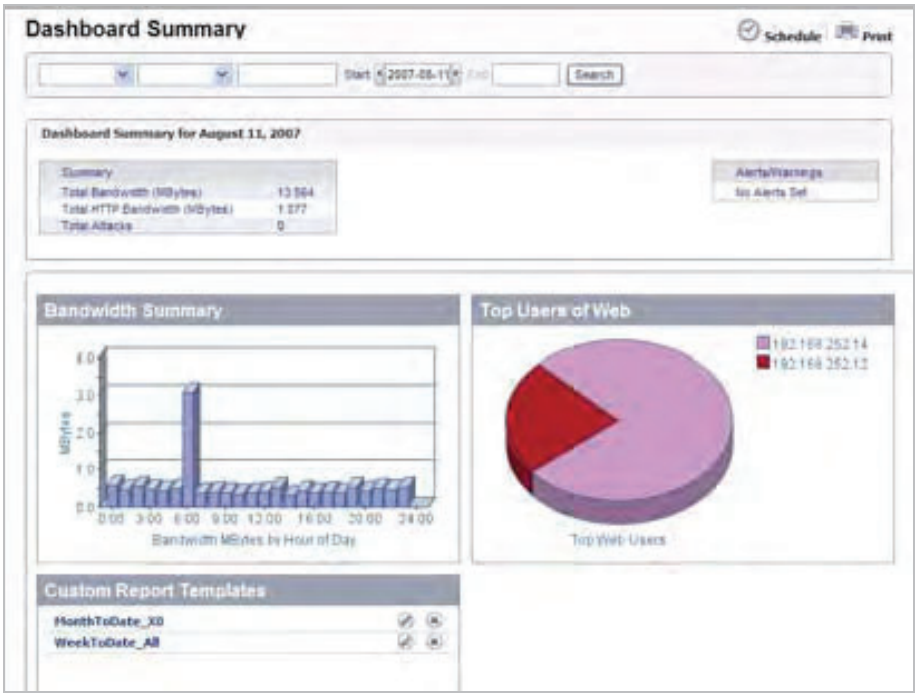
To view the Dashboard Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1404
Keysight, EX-1045
Page 1404

3. Expand the **Dashboard** tree and click **Summary**.




4. The tables at the top of the page display the totals, using megabytes for the bandwidth totals.
5. The graphical display breaks down the information as follows:
   - **Bandwidth**—shown by group when viewed at global level. At the unit level, the bandwidth is shown per hour.
   - **HTTP Bandwidth**—at the unit level, this is shown as a pie chart with eight slices. The top seven Web users by IP address are each shown as a slice, with all other HTTP bandwidth combined in the eighth slice.
   - **Attacks Events**—at the global level, both attack events and virus attack attempts are shown per group. At unit level, these are shown per hour (not pictured).
   - **Custom Report Templates**—your “favorites” list of saved custom report templates. See “Using Custom Reports on UTM Appliances” on page 163.

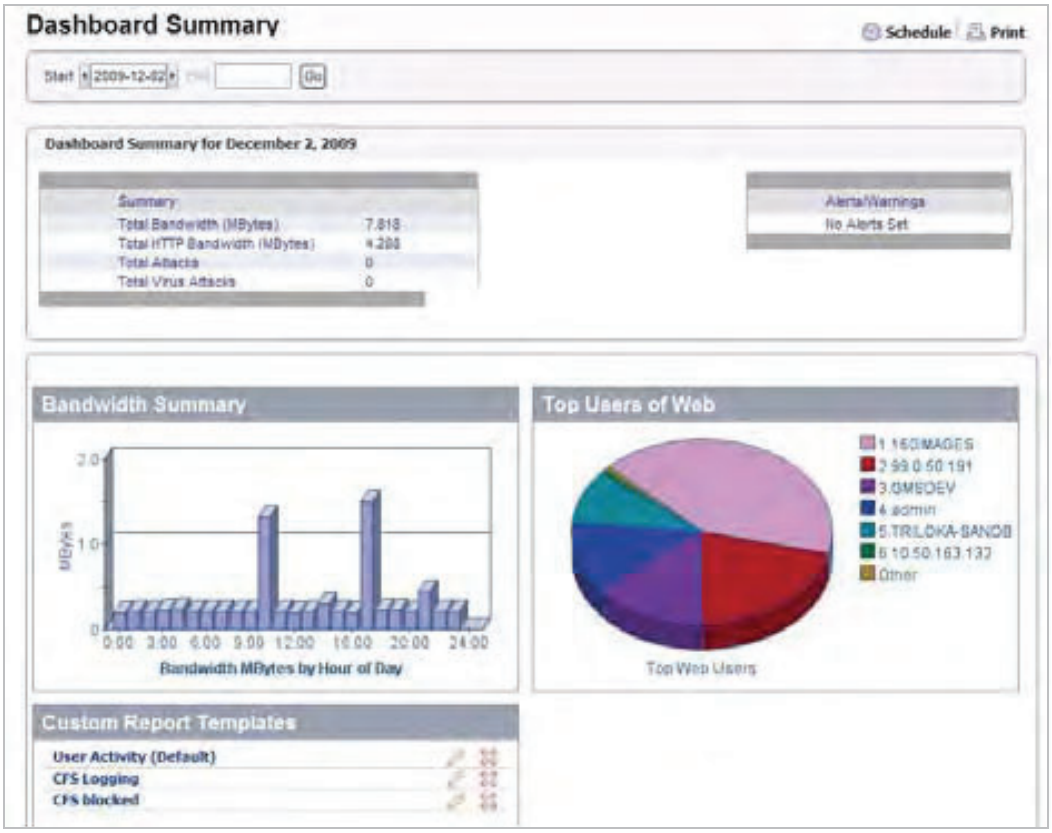
     You can click the Edit icon next to the template on this page to edit the template in the Custom Report page and save it using the Save Template button. To delete the template, click the Delete icon.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1405
Keysight, EX-1045
Page 1405

# Viewing Custom Reports on the Dashboard

SonicWALL ViewPoint provides access to your saved Custom Report templates on the Dashboard > Summary page for the appliance. The template must have been previously created and saved for the same appliance on the Custom Report > Internet Activity or Custom Report > Website Filtering page.




When you click on a saved template, the detailed report page is displayed in Full Mode with the same categories in the same order as in the template that you saved. In the report page, the Print, PDF, and Excel icons are available, along with the pagination controls. There is no link to Split Mode and no Save Template button since this template is already saved.

You can also configure or delete a saved template from the Dashboard > Summary page.

To access a custom report from the Dashboard:

1. Select a unit for which Log Viewer is enabled, and then navigate to **Dashboard > Summary**.
2. Locate the box labeled **Custom Report Templates**. All saved templates for this appliance are listed in the box.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1406
Keysight, EX-1045
Page 1406

3. Do one of the following:

- To generate a Custom Report, click a saved template in the **Custom Report Templates** box.
- To configure a saved template, click the Configure icon for that template, make the desired changes, and then click **OK**. For configuration instructions, see "Using Custom Reports on UTM Appliances" on page 163.
- To delete a saved template, click the Delete icon for that template and then click **OK** in the confirmation dialog box.

# Viewing the Security Dashboard Report

The Security Dashboard report shows two types of reports:

- An Individual Appliance Report that displays a summary of attacks detected by the local SonicWALL security appliance.
- A Global Report that displays a summary of threat data received from all SonicWALL security appliances worldwide.

The Dashboard > Security Dashboard screen is available at the global level, but not at unit level for SonicWALL CSM Series appliances.

To view the Security Dashboard report, perform the following steps:

1. Click the **Reports** tab.
2. Select the global icon, a group, or a SonicWALL appliance.
3. Expand the **Dashboard** tree and click **Security Dashboard**. The Security Dashboard page displays.

***Figure 8 Security Dashboard Page***

4. At the top of the screen, select either the **Global** radio button or, for reporting at unit level, select the radio button that is labeled with the unit's MAC address. Select Global to display a summary of attacks caught by SonicWALL appliances worldwide. Select the unit's MAC address to see results only for attacks through this unit. At all levels, the categories charted include the following:
   - Viruses Blocked by SonicWALL Network
   - Intrusions Prevented by SonicWALL Network
   - Spyware Blocked
   - Multimedia (IM/P2P) Detected/Blocked



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1407
Keysight, EX-1045
Page 1407

For each of these, the report includes the results over time for the top ten.

5. Optionally select the period of time for the report from the drop-down box at the top right of each graphical display. At the unit level, you can select only the **Last 21 days**. At the global or group level, you can select from:
   - Last 12 Hours
   - Last 14 Days
   - Last 21 Days
   - Last 6 Months

# Using Custom Reports on UTM Appliances

Custom Reports are available at the unit level for appliances visible on the UTM tab. Log Viewer must be enabled for the appliance. For information about enabling Log Viewer, see “Viewing the Log” on page 290.

When configuring a Custom Report on the Internet Activity or Website Filtering page, the Template Section acts as a query builder. You select the criteria for the report that you want, and SonicWALL ViewPoint uses your input to query the raw syslog database for the information, and then outputs the report. The Template Section consists of two parts: the Date/Time section and the Report Layout section.

After building your query in the Template Section and clicking the **Generate Report** button, the report is displayed in the Report Section. The Report Section is displayed in the lower half of the page, under the Template Section; this layout is called Split Mode. You can easily toggle between Split Mode and Full Mode. Full Mode can be used to display only the Template Section or only the Report Section in a full page view.

The Report Section displays the report and provides controls for pagination, printing, and exporting the report in PDF or CSV format. You can also click the **Save Template** button in this section if you want to save the settings for this report as a template for reuse later.

See the following sections for detailed information:


- “Toggling Between Split Mode and Full Mode” on page 164
- “Configuring the Date and Time for Custom Reports” on page 166
- “Configuring the Report Layout and Generating the Report” on page 168
- “Generating the Custom Report” on page 176
- “Viewing a Custom Report” on page 177



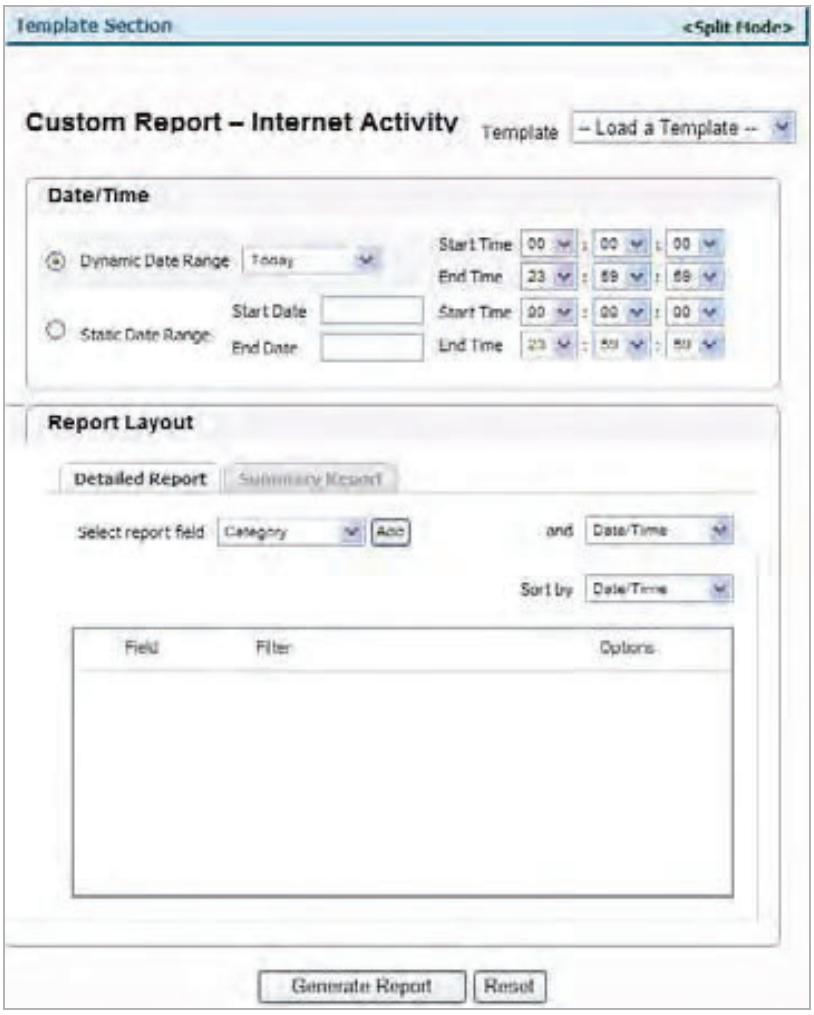

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1408
Keysight, EX-1045
Page 1408

- "Printing a Page or Exporting the Report as a PDF or CSV File" on page 179
- "Saving the Report Template" on page 180


# Toggling Between Split Mode and Full Mode

The Custom Report page contains two main sections, the Template Section and Report Section, which can be displayed together or independently depending on the mode.

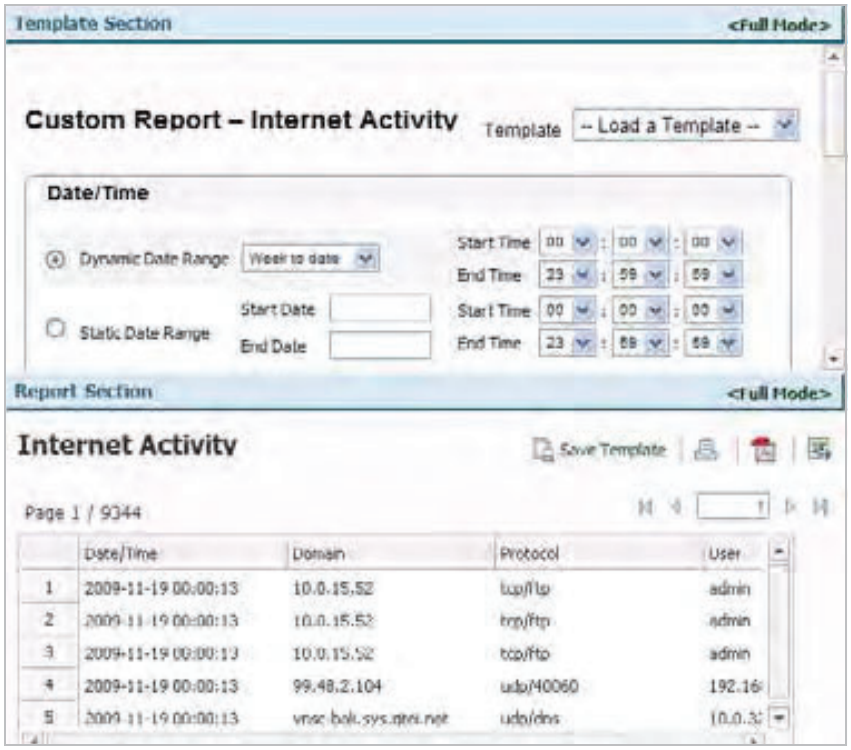
When the Custom Report page is initially displayed for a selected appliance, the Template Section is displayed in Full Mode. Split Mode is available, but the Report Section displays no data until a report has been generated. The Custom Report > Internet Activity page with the Template Section displayed in Full Mode is shown below.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1409
Keysight, EX-1045
Page 1409

After generating a report, the page automatically changes to Split Mode and displays the report settings in the Template Section in the top half of the page and the report results in the Report Section in the lower portion. The Template Section and Report Section displayed in Split Mode is shown below.




At any time, you can change to Full Mode if you want to display either the Template Section or the Report Section individually. From Full Mode, you can easily change back to Split Mode.

To toggle between Split Mode and Full Mode:

1. Select a unit for which Log Viewer is enabled, and then navigate to the **Custom Report** page.
2. On a page that is currently displayed in Full Mode, to change the view to Split Mode click the **<Split Mode>** button at the right side of the section heading.
3. On a page that is currently displayed in Split Mode, do one of the following to change to a Full Mode display of either the Template Section or the Report Section:
   - Click the **<Full Mode>** button to the right of the Template Section heading.
   - Click the **<Full Mode>** button to the right of the Report Section heading.

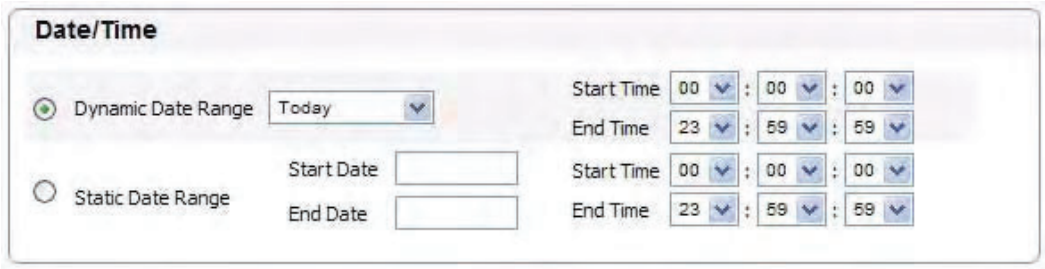


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1410
Keysight, EX-1045
Page 1410

# Configuring the Date and Time for Custom Reports

At the top of the Template Section of the Custom Report page, the Date/Time region provides a way to designate the time period to use when generating the report. You can select either a Dynamic Date Range or a Static Date Range.

Both the Dynamic Date Range and the Static Date Range provide Start Time and End Time settings. By using the Start Time and End Time fields, you can specify the exact hour, minute, and second for both the beginning and the end of the period for the report. When a start and end time is specified for a date range containing multiple days, the start/end times are applied to each day of the period when analyzing data for the report. The default is to include data for the full 24 hours in each day of the date range.




## Dynamic Date Range

The **Dynamic Date Range** selection allows you to select from four date ranges and to specify the exact starting and ending times on the days in the selected date range for the log data to be used for the report.

For the **Dynamic Date Range**, you can select from the following four date choices:

- **Today** – Uses log data from the current date, beginning just after midnight
- **Yesterday** – Uses log data from just after midnight of the previous day, up to and including the most recent log message from the current date
- **Week to Date** – Uses log data from the current date, plus the seven preceding days
- **Month to Date** – Uses log data from the same date as the current date in the previous month, up to and including the most recent log message from the current date

When generating a report with a template containing a dynamic date range setting, the dates used when referencing the log data are relative to the current date. Thus, two reports generated from the same template on different days will provide different results.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1411
Keysight, EX-1045
Page 1411

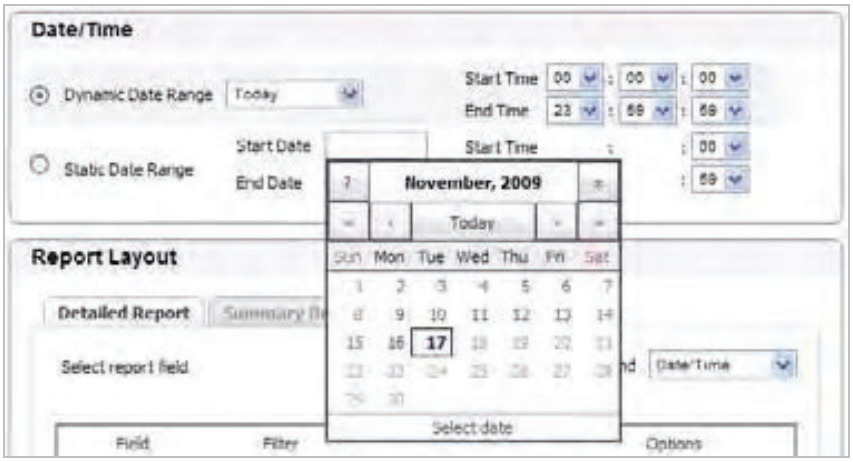
To select a Dynamic Date Range:

1. Select a unit for which Log Viewer is enabled, and then navigate to the page under **Custom Report** for the report type you want.
2. In the Template Section under Date/Time, select the **Dynamic Date Range** radio button.
3. In the drop-down list, select **Today**, **Yesterday**, **Week to Date**, or **Month to Date**.
4. For the **Start Time**, select the hour, minute, and second from the drop-down lists in the Dynamic Date Range row. These settings specify the earliest data to be included in the report, for each day of the date range.
5. For the **End Time**, select the hour, minute, and second from the drop-down lists. These settings specify the most recent data to be included in the report, for each day of the date range.
6. To change the settings back to the defaults, click **Reset** at the bottom of the Template Section. Note that this will change the Report Layout region as well as the Date/Time region back to default settings.

## Static Date Range

The **Static Date Range** selection allows you to specify the exact dates, starting, and ending times on the days in the selected date range for the log data to be used for the report. You can specify a single date or a date range, and indicate the exact hour, minute, and second for both the beginning and the end of the daily period for the report.

A popup calendar makes it easy to select the Start Date and End Date for the date range.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1412
Keysight, EX-1045
Page 1412

To specify a Static Date Range:

1. Select a unit for which Log Viewer is enabled, and then navigate to the page under **Custom Report** for the report type you want.
2. In the Template Section under Date/Time, select the **Static Date Range** radio button.
3. Click the **Start Date** field to access the pop-up calendar.
4. Use the navigation arrows near the top of the calendar to change the year or month. Click the **<<** button to move to the previous year, or hold the button to select from a list of years. Click the **>>** button to move to the next year, or hold the button to select from a list of years. Similarly, click the **<** or **>** to move back or ahead by one month, or hold the button to select from a list of months.
5. Click the desired start date in the calendar. This adds the date to the **Start Date** field and closes the calendar.
6. Click the **End Date** field to access the pop-up calendar.
7. Use the navigation arrows near the top of the calendar to change the year or month.
8. Click the desired end date in the calendar. This adds the date to the **End Date** field and closes the calendar.
9. For the **Start Time**, select the hour, minute, and second from the drop-down lists in the Static Date Range row. These settings specify the earliest data for each day in the date range to be included in the report.
10. For the **End Time**, select the hour, minute, and second from the drop-down lists. These settings specify the most recent data for each day in the date range to be included in the report.
11. To change the settings back to the defaults, click **Reset** at the bottom of the Template Section. Note that this will change the Report Layout region as well as the Date/Time region back to default settings.

# Configuring the Report Layout and Generating the Report

Located in the Template Section of the Custom Report page below the Date/Time region, the Report Layout region provides a way to specify the type of data to include, and the format of the report. The Report Layout region has a Detailed Report tab and a Summary Report tab. The report appearance and the way information is organized is quite different between a Detailed Report and a Summary Report.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1413
Keysight, EX-1045
Page 1413

The Detailed Report tab contains a list of data categories that you can add as report fields, and allows you to specify query values for each. The categories you select will appear as column headings in the report.

The Summary Report tab allows you to structure a report showing the top elements of Internet Activity or Website Filtering. You can select the number of top elements, what to base the comparisons on, and the two data categories to evaluate when determining the top elements. The generated report provides graphical output that you can click to drill down for detailed information.

For more information about each of these Report Layout tabs, see the following sections:

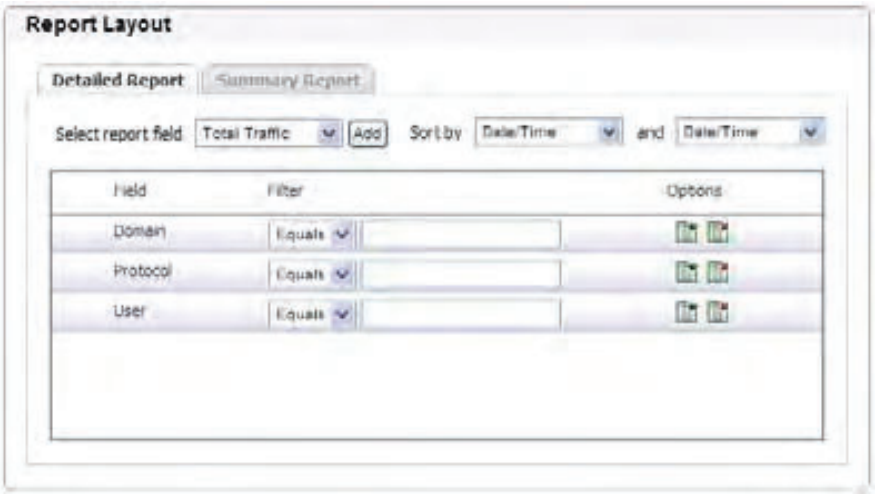
- “Detailed Reports” on page 169
- “Summary Reports” on page 173

For information about the Filter operators, see the following section:

- “Filter Operators” on page 175

## Detailed Reports

The Detailed Report tab is the default view in the Report Layout region.




For a UTM Internet Activity report, the **Select Report Field** drop-down list contains eight data categories that you can add as column headings in the report. The categories are:

- **Full URL** – Adds a column containing the full URL of each Web site visited
- **Category** – Adds a column containing the category of each site visited, such as Gambling or Adult/Mature Content



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1414
Keysight, EX-1045
Page 1414

- **Domain** – Adds a column containing the domain name of each site visited
- **Protocol** – Adds a column containing the protocol used by the traffic
- **Received Traffic**– Adds a column containing the number of bytes received from the visited site
- **Transmitted Traffic** – Adds a column containing the number of bytes transmitted to the site
- **Total Traffic** – Adds a column containing the total number of bytes received and transmitted
- **User** – Adds a column containing the user ID

For a UTM Website Filtering report, the **Select report field** drop-down list contains four data categories that you can add as column headings in the report. The categories are:

- **Full URL** – Adds a column containing the full URL of each logged Web site
- **Category** – Adds a column containing the category of each logged site, such as Gambling or Adult/Mature Content
- **Domain** – Adds a column containing the domain name of each logged Web site
- **User** – Adds a column containing the user ID

To include a field in the report, select a choice from the list and then click **Add**. When you click **Add**, a row is populated in the table below, which has three column headings: Field, Filter, and Options.



**Note** When you place your mouse cursor over the row, under the Field heading, the cursor changes to a “move” cursor. You can drag and drop the rows to rearrange the column ordering in the final report.

In the Filter column, two fields are displayed: an operator field and an input field. The operator field is a drop-down list containing the operator choices for the selected report field. See “Filter Operators” on page 175 for a description of each operator. The input field can be a drop-down list or a standard input field, depending on the selected report field.

The operators and input fields are defined in Table 5 for each report field.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1415
Keysight, EX-1045
Page 1415

*Table 5 Operators and Input Fields for Each Data Type*

| Data Type | Operators | Input Field |
|---|---|---|
| Category | Equals | The input field is a drop-down list containing an alphabetized list of all the content filtering categories, such as Adult/Mature Content, Gambling, Military, etc. Leave the default of All in the input field if you choose not to filter by a certain category. |
| Destination IP | Equals<br>Starts with<br>Ends with<br>Contains | The input field is a standard input field where you can type in the numbers to match, such as 192 or 10.25. Leave the input field blank if you choose not to filter by a certain destination IP address. |
| Domain | Equals<br>Start with<br>End with<br>Contains | The input field is a standard input field where you can type in the domain to match, such as sonicwall.com. Leave the input field blank if you choose not to filter by a certain domain. |
| Full URL | Equals<br>Start with<br>End with<br>Contains | The input field is a standard input field where you can type in the URL to match, such as: `http://www.funnyyoutubevideo.com/funniest.html`<br>Leave the input field blank if you choose not to filter by a certain URL. |
| Protocol | Equals<br>Start with<br>End with<br>Contains | The input field is a standard input field where you can type in the protocol to match, such as FTP. Leave the input field blank if you choose not to filter by a certain protocol. |
| Received Traffic | =<br>><br>>=<br><<br><=<br>!= | The input field is a standard input field where you can type in the number of bytes to match or compare to. Leave the input field blank if you choose not to filter by a certain amount of traffic. |
| Source IP | Equals<br>Starts with<br>Ends with<br>Contains | The input field is a standard input field where you can type in the numbers to match, such as 192 or 10.25. Leave the input field blank if you choose not to filter by a certain source IP address. |
| Total Traffic | =<br>><br>>=<br><<br><=<br>!= | The input field is a standard input field where you can type in the number of bytes to match or compare to. Leave the input field blank if you choose not to filter by a certain amount of traffic. |



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1416
Keysight, EX-1045
Page 1416

| Data Type | Operators | Input Field |
|---|---|---|
| Transmitted Traffic | =<br>><br>>=<br><<br><=<br>!= | The input field is a standard input field where you can type in the number of bytes to match or compare to. Leave the input field blank if you choose not to filter by a certain amount of traffic. |
| User | Equals<br>Start with<br>End with<br>Contains | The input field is a standard input field where you can type in the user ID to match. Leave the input field blank if you choose not to filter by a certain user. |

In the Options column, two icons are displayed: an Eye  and an X  . You can click the Eye to toggle whether the report field on that row will be displayed in the final report. This allows you to filter the report results based on the selected report field and related filter value, but not display the field as a column. When you click on the Eye icon within a row, the eye closes  to show that this field will not be displayed in the final report. The filter value will still be used to filter results from the raw syslog database to apply towards the report.

For example, you might specify the following Field/Operator/Filter Value: **Protocol/=/http**. It would make sense to click the Eye icon to disable the Protocol field from being shown in the report, since it would always just be “http” and would not add any interesting information to the final report. Contrast this with simply specifying the Protocol field and leaving the Filter Value blank, in which case you would want to enable the Eye so that this column would appear in the report showing a variety of protocols such as udp/dns, tcp/http, udp/ntp, or numbered protocols such as udp/389 (the LDAP protocol) or tcp/445 (MS Server Message Block (SMB) file sharing).

Clicking the X icon under Options deletes the selected report field from the table, so it will not be used to generate the report results nor will it be displayed in the report. Use the X icon instead of the Eye when you do not choose to filter the report results based on the field.

The Detailed Report tab also contains the Sort By drop-down list. The list contains the Date/Time option and any other report fields that you have selected from the eight data types. The choice you select will be used to order the results in the report from the first page to the last. The selection in the left drop-down list is used for the first sorting, then the selection in the right drop-down list is used to sort and group the entries within each group resulting from the the first sorting.



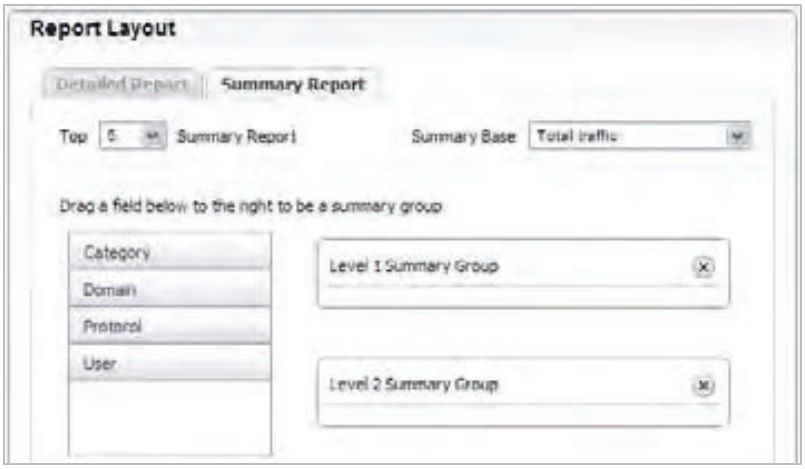
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1417
Keysight, EX-1045
Page 1417

To configure a detailed report:

1. Select a unit for which Log Viewer is enabled, and then navigate to the page under **Custom Report** for the report type you want.
2. In Report Layout region of the Template Section of the Custom Report page, select the **Detailed Report** tab.
3. In the **Select report field** drop-down list, select a data type to include in the report, and then click **Add**. A row for this field is populated in the table below. Repeat this step to add other fields.
4. Optionally select an operator from the drop-down list under **Filter** in a table row, and type in or select an input value to be matched when the database is queried. Repeat this step for other rows to add filter values for those fields.
5. To prevent a field from appearing in the final report, click the **Eye** icon in that row so that the eye appears closed. To allow the field to be displayed in the report, click the closed **Eye** icon to return it to normal appearance.
6. To delete a field from the table, click the **X** icon in that row.
7. To sort the report pages by a different field than the default of Date/Time, select the desired field from the **Sort by** drop-down list.
8. To change the settings back to the defaults, click **Reset** at the bottom of the Template Section. Note that this will change the Date/Time region and the Report Layout region back to default settings.

## Summary Reports

The Summary Report tab is available in the Report Layout region of the Template Section.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1418
Keysight, EX-1045
Page 1418

The Top drop-down list provides selections for the number of entries to display in the report. For example, if the User field is selected below as a Summary Group, and 5 is selected in the Top drop-down list, the report will provide entries for the top five users. For all Custom Reports, available numbers in the Top drop-down list are 5, 10, 20, 50, and 100.

The Summary Base drop-down list offers a selection of traffic types that will be used to determine the top usage for the selected field. The Summary Base choices vary as follows depending on the type of Custom Report:

- For a UTM Internet Activity report, the Summary Base choices are Total traffic, Received traffic, or Transmitted traffic.
- For a UTM Website Filtering report, the only Summary Base choice is Filtered Items.

Below the Top and Summary Base fields, you can create one or two Summary Groups from the choices listed on the left side. The Summary Groups choices vary as follows depending on the type of Custom Report:

- For a UTM Internet Activity report, the choices are Total traffic, Received traffic, or Transmitted traffic.
- For a UTM Website Filtering report, the choices are Category, Domain, or User.

To select a field for a Summary Group, simply drag and drop the desired field from the list to either the Level 1 Summary Group or Level 2 Summary Group boxes. When the field name is dragged to one of these, the operator drop-down list and filter input value field are displayed, allowing you to specify values to match when the data is searched. See “Filter Operators” on page 175 for a description of each operator.

Either the Level 1 Summary Group field or the Level 2 Summary Group field can be used alone; the resulting report will look the same in both cases.

When both the Level 1 and Level 2 Summary Group fields are populated, the report will display the top entries for the Level 2 field for each of the top entries for the Level 1 field. For example, if User is dragged to the Level 1 Summary Group and Domain is dragged to the Level 2 Summary Group, and 5 is selected in the Top drop-down list, the generated report will display the top five domains visited by each of the top five users.

To configure a summary report:

1. Select a unit for which Log Viewer is enabled, and then navigate to the page under **Custom Report** for the report type you want.
2. In Report Layout region of the Template Section of the Custom Report page, select the **Summary Report** tab.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1419
Keysight, EX-1045
Page 1419

3. In the **Top** drop-down list, select the number of entries to be displayed in the report.
4. In the **Summary Base** drop-down list, select one of the choices to use when determining which are the top elements in the selected field.
5. To specify the field for the **Level 1 Summary Group**, click and drag the desired field from the list on the left to the **Level 1 Summary Group** field, and then release your mouse button to drop the field into position. The filter operator and input field are displayed next to the field name.




6. To specify the field for the **Level 2 Summary Group**, click and drag the desired field from the list on the left to the **Level 2 Summary Group** field, then release your mouse button to drop the field into position. The filter operator and input field are displayed next to the field name.
7. To specify a filter operator and filter value for a Summary Group, select the operator from the drop-down list next to the field and type a filter value into the input field to the right of the operator.
8. To change the settings back to the defaults, click **Reset** at the bottom of the Template Section. Note that this will change the Date/Time region as well as the Report Layout region back to default settings.

## Filter Operators

When configuring the Report Layout on either the Detailed Report tab or the Summary Report tab, you can specify filter values to be matched in the database during report generation. Depending on the selected field type, text string or numeric, several filter operators are available. The filter operators are used with a filter input value to determine which data should be included in the report.

The operators are defined as shown in Table 6.

***Table 6 Filter Operators***

| Operator | Definition |
|---|---|
| Equals | Only data that exactly matches the filter input text will be included in the report |
| Start with | Data that begins with the input text will be included in the report |



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1420
Keysight, EX-1045
Page 1420

| Operator | Definition |
|---|---|
| End with | Data that ends with the input text will be included in the report |
| Contains | Data that contains the input text will be included in the report |
| = | Only data that exactly matches the filter input numerical value will be included in the report |
| > | Data values that are greater than the input numerical value will be included in the report |
| >= | Data values that are greater than or equal to the input numerical value will be included in the report |
| <= | Data values that are less than or equal to the input numerical value will be included in the report |
| < | Data values that are less than the input numerical value will be included in the report |
| != | Data values that are not equal to the input numerical value will be included in the report |

# Generating the Custom Report

The Generate Report button at the bottom of the Template Section is used to create the report. Before clicking Generate Report, use the Template Section to specify the time period for the report and the contents and layout of the report.



**Note** Custom Reports are available at the unit level and Log Viewer must be enabled for the appliance. For information about enabling Log Viewer, see “Viewing the Log” on page 290.

To generate a custom report:

1. Select a unit for which Log Viewer is enabled, and then navigate to the page under **Custom Report** for the report you want.
2. In the **Date/Time** region of the Template Section, specify the time period that the report will cover. For detailed information and instructions, see “Configuring the Date and Time for Custom Reports” on page 166.



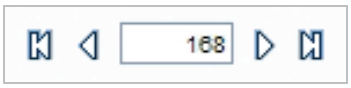
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1421
Keysight, EX-1045
Page 1421

3. In the **Report Layout** region of the Template Section, specify the contents and appearance of the report. For detailed information and instructions, see “Configuring the Report Layout and Generating the Report” on page 168.
4. Click **Generate Report** to create the report using the specified configuration.

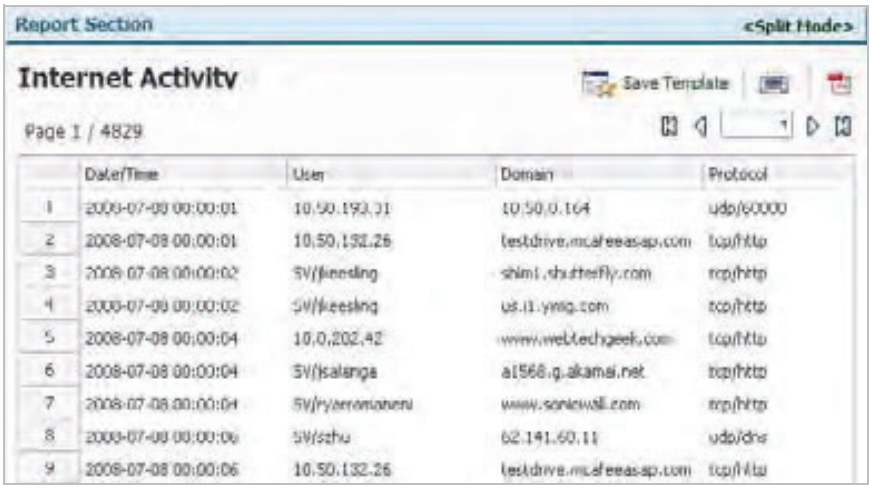
# Viewing a Custom Report

After you click Generate Report, the Report Section is displayed in Split Mode in the lower half of the main window, even if you previously were in Full Mode for the Template Section.

Pagination controls are displayed at the upper right of the report, just below the Save Template button and the printer, PDF, and Excel icons. Navigation buttons are provided to take you to the first page, next page, previous page, and last page, or you can specify an exact page number in the field.



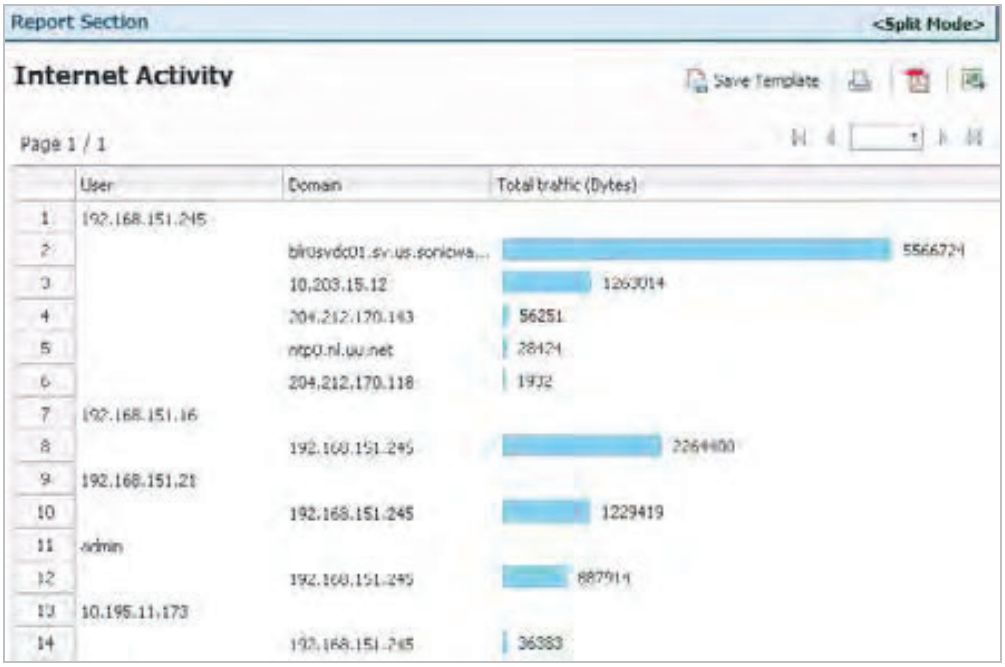

In a Detailed Report, shown below, the selected report fields are displayed as column headings. You can click on any column heading to sort that page by the values in the column that you click. Click again to toggle between ascending and descending order on that page. When you navigate away from that page and then come back using the pagination controls, the page reverts to the original sorting order as specified in the **Sort by** field of the Template Section before generating the report.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1422
Keysight, EX-1045
Page 1422

In a Summary Report, the Report Section displays the traffic volume as horizontal bar charts. This lets you see the information at a glance, such as who consumed the most bandwidth and which domains they visited the most.

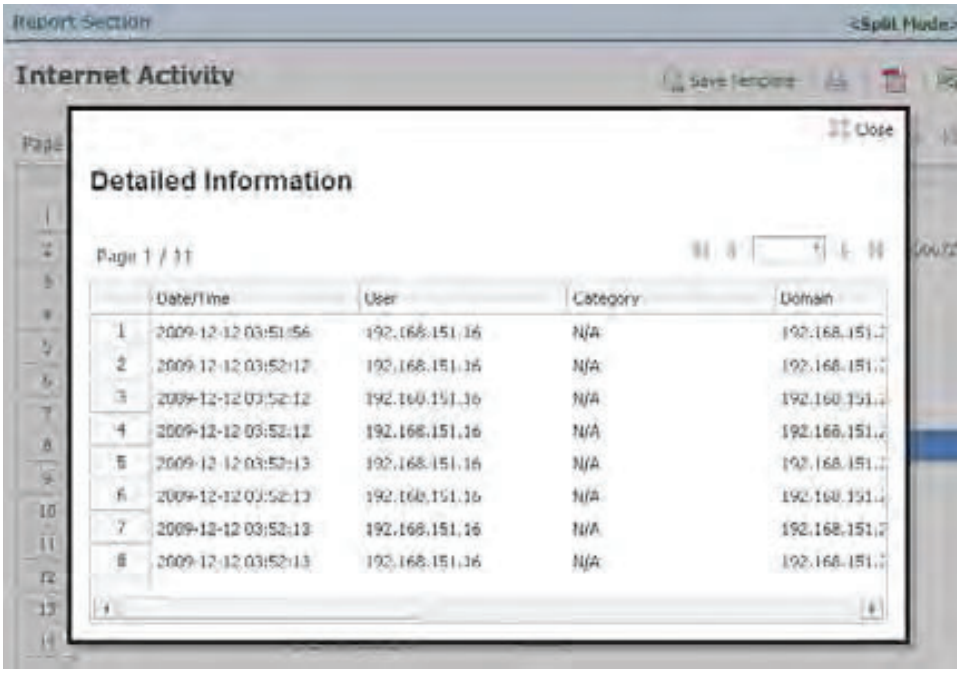



You can click on a bar in the chart to pop up detailed information, just like the detailed report with all of the columns for all fields. The report lists details about this Summary Group field only. For example, in the Internet Activity report, if the Summary Group contains the User field and you click on a bar for one of the top users, the report displays the date and time of all Internet activity for the user, and includes data for every field available for detailed reports. A scroll bar is provided along the bottom of the Detailed Information window to allow viewing of all eight fields plus the date and time column.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1423
Keysight, EX-1045
Page 1423

The Detailed Information window is shown below.




# Printing a Page or Exporting the Report as a PDF or CSV File

To print the current page of the report, click the printer icon at the top of the Report Section. Your normal print dialog box pops up. This prints only the page that is currently displayed.

To export the entire report in PDF format, click the PDF icon at the top of the Report Section. A PDF file is generated showing the report results in table format.

To export the entire report in Microsoft Excel Comma Separated Value (CSV) format, click the Excel icon at the top of the Report Section. A CSV file is generated showing the report results in spreadsheet format.

The PDF can contain a maximum of 10,000 records. If your report contains more than 10,000 records, you can use the Static Date Range fields to adjust the dates and regenerate the report to shorten its length. You can save the PDF or CSV file using any filename and location.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1424
Keysight, EX-1045
Page 1424

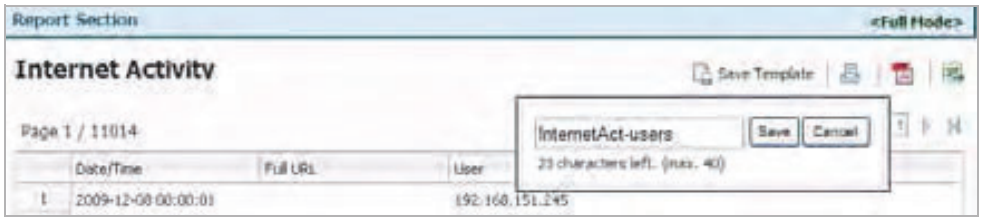
## Saving the Report Template

After generating the report, you can save the settings for this report as a template for reuse. You can select the saved template from the Template Section or from the Dashboard > Summary page at a later time, and use it to generate a report using the same settings. For information about using the template on the Dashboard > Summary page, see "Troubleshooting Reports" on page 156.

The template is saved for the currently selected appliance and for the specific user. The saved template will not be available for other appliances or for other users.

To save the report template:

1. In the Report Section in the upper right corner, click the **Save Template** button.




2. In the popup dialog box, type in a descriptive name for the template, up to 40 characters. The number of remaining characters allowed in the name is displayed below the input field and changes as you type.
3. Click **Save**. If you are in a Full Mode display of the Report Section, you can verify that the template has been saved by changing back to Split Mode and viewing the contents of the **Template** drop-down list.

SonicWALL ViewPoint provides access to your saved Custom Report templates on the Dashboard > Summary page for the appliance. See "Viewing Custom Reports on the Dashboard" on page 161.

# Viewing Bandwidth Reports

Bandwidth reports display the amount of data transferred through one or more selected SonicWALL appliances. These reports include the cost of consumed network bandwidth per 100 megabytes transferred through the selected appliances.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1425
Keysight, EX-1045
Page 1425

Bandwidth reports are an ideal starting point for viewing overall bandwidth usage. You can view bandwidth usage view by hour, day, or over a period of days. Additionally, you can view the top users of bandwidth.

From this information, you can determine network strategies. For example, if you need more bandwidth, you might need to upgrade network equipment, or you might simply need to curtail the bandwidth usage of a few employees.



**Note** All reports appear in the appliance's time zone.

Select from the following:


- "Viewing the Bandwidth Summary Report" on page 181
- "Viewing the Top Users of Bandwidth" on page 183
- "Viewing Bandwidth Usage Over Time" on page 185
- "Viewing the Top Users of Bandwidth Over Time" on page 187


# Viewing the Bandwidth Summary Report

The Bandwidth Summary report contains information on the amount of traffic handled by a SonicWALL appliance during each hour of the specified day, or at the global level, for all SonicWALL appliances for the day.

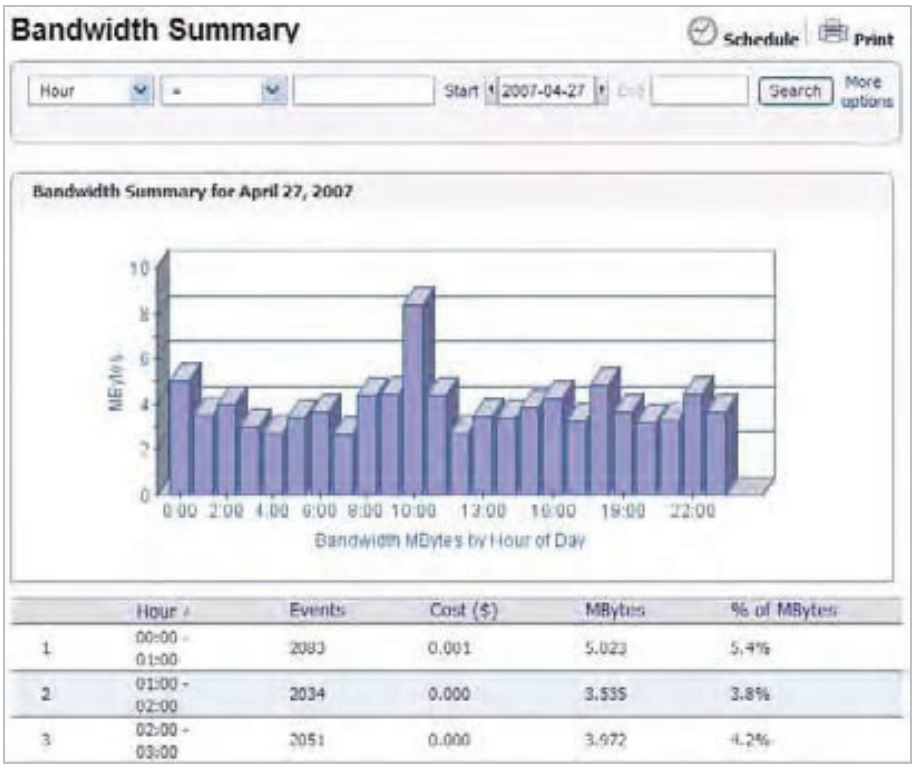
To view the Bandwidth Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1426
Keysight, EX-1045
Page 1426

3. Expand the **Bandwidth** tree and click **Summary**. The Summary page displays.




4. The bar graph displays the amount of bandwidth transferred during each hour of the day.
5. The table contains the following information:
   - **Hour**—when the sample was taken.
   - **Events**—number of events or “hits.”
   - **Cost ($)**—amount of the expense per 100 megabytes. You can configure this in the **Cost Per Mega Byte Bandwidth Use** field in the Console > Reports > Summarizer screen.
   - **MBytes**—number of megabytes transferred.
   - **% of MBytes**—percentage of megabytes transferred during this hour, compared to the day. For example, if 1000 megabytes of data was transferred during the day and 100 megabytes was transferred at the 12:00 time period, the **% of MBytes** field will display 10%.
6. The ViewPoint Reporting Module shows yesterday’s report. To change the date of the report and other settings, click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1427
Keysight, EX-1045
Page 1427

- **Display Type**: **Chart and Table**, or **Table Only**
- **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
- Select the **Source** and **Destination** interfaces to view
- If you want to track bandwidth usage in both directions, select the **Bi-directional** check box.

See "Managing Report Settings" on page 154.

8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected day.



**Note** These settings will stay in effect for all summary reports during your active login session.

# Viewing the Top Users of Bandwidth

The Top Users report displays the users who used the most bandwidth on the specified date and the correlating expense.

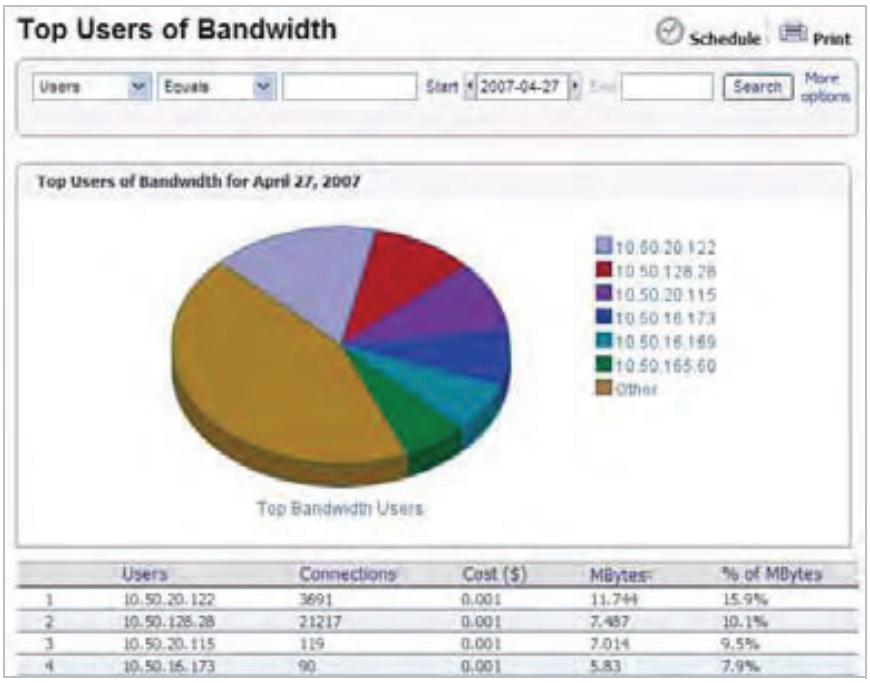
To view the Top Users report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1428
Keysight, EX-1045
Page 1428

3. Expand the **Bandwidth** tree and click **Top Users**. The Top Users page displays.




4. The pie chart displays the percentage of bandwidth transferred by each user.
5. The table contains the following information:
   - **Users**—the IP address of the user.
   - **Connections**—number of events or “hits.”
   - **Cost ($)**—amount of the expense per 100 megabytes. You can configure this in the **Cost Per Mega Byte Bandwidth Use** field in the Console > Reports > Summarizer screen.
   - **MBytes**—number of megabytes.
   - **% of MBytes**—percentage of megabytes transferred by this user, compared to all users. For example, if 1000 megabytes of data was transferred during the day and 200 megabytes was transferred by the top user, the **% of MBytes** field will display 20%.
6. By default, the ViewPoint Reporting Module shows yesterday’s report, a pie chart, and the ten top users. To change the date of the report and other settings, click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1429
Keysight, EX-1045
Page 1429

7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. To display a limited number of users, use the Search Bar fields.



**Note** The search bar fields use pattern matching with operators such as "contains". For example, "john" will match john_smith, john42, or big_john.

9. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected day.



**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Bandwidth Usage Over Time

The Bandwidth Over Time report displays the daily amount of traffic and the total daily expense for consumed network bandwidth handled by a SonicWALL appliance or a group of SonicWALL appliances for the specified time period.

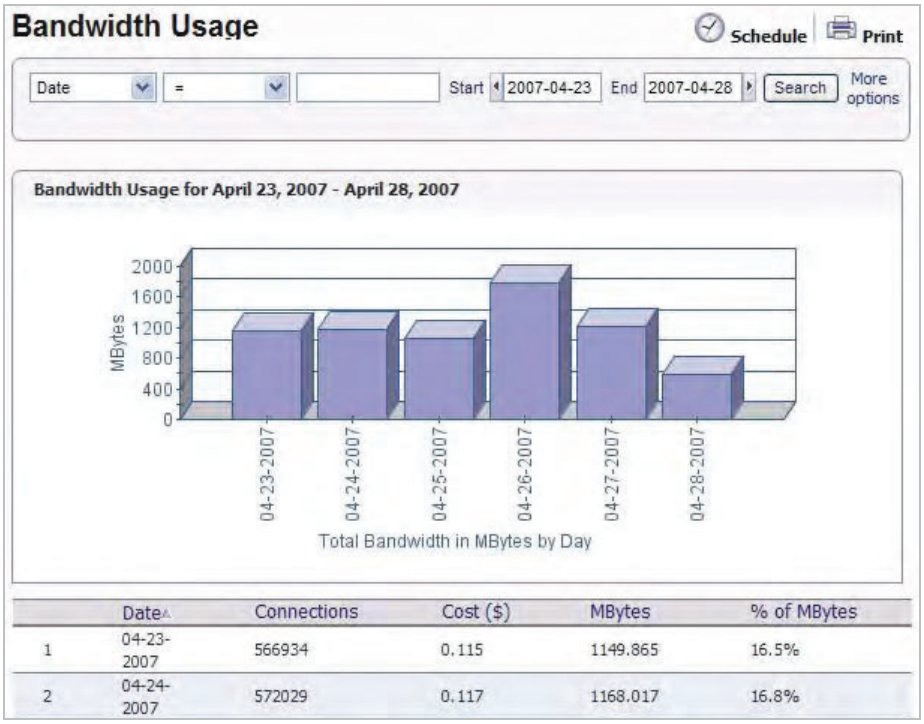
To view the Bandwidth Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1430
Keysight, EX-1045
Page 1430

3. Expand the **Bandwidth** tree and click **Over Time**. The Over Time page displays.




| | Date | Connections | Cost ($) | MBytes | % of MBytes |
|---|---|---|---|---|---|
| 1 | 04-23-2007 | 566934 | 0.115 | 1149.865 | 16.5% |
| 2 | 04-24-2007 | 572029 | 0.117 | 1168.017 | 16.8% |

4. The bar graph displays the amount of bandwidth transferred during each day of the specified time period.
5. The table contains the following information:
   - **Date**—when the sample was taken.
   - **Connections**—number of hits.
   - **Cost ($)**—amount of the expense per 100 megabytes. You can configure this in the **Cost Per Mega Byte Bandwidth Use** field in the Console > Reports > Summarizer screen.
   - **MBytes**—number of megabytes transferred.
   - **% of MBytes**—percentage of megabytes transferred during this day, compared to the time period. For example, if 100,000 megabytes of data was transferred during the time period and 25,000 megabytes was transferred on one day, the **% of MBytes** field will display 25%.
6. To change the date of the report and other settings, use the Search Bar and click the **Start** or **End** fields to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1431
Keysight, EX-1045
Page 1431

– **Chart Type**: **Area**, **Bar** or **Plot** chart

See “Managing Report Settings” on page 154.

8. When you are finished, click **Search.** The ViewPoint Reporting Module displays the report for the selected date range.

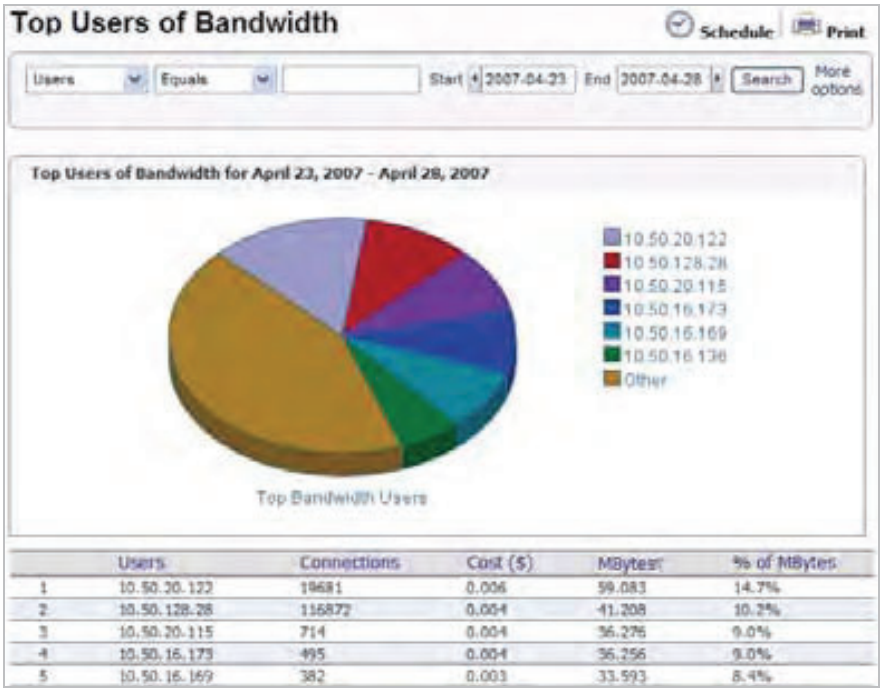
**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing the Top Users of Bandwidth Over Time

The Top Users Over Time report displays the users who used the most bandwidth and accumulated the highest cost during the specified date range. This report is available at the unit level.

To view the Top Users Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Bandwidth** tree and click **Top Users Over Time**. The Top Users Over Time page displays.




4. The pie chart displays the percentage of bandwidth transferred by each user.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1432
Keysight, EX-1045
Page 1432

5. The table contains the following information:
   - **Users**—the IP address of the user.
   - **Connections**—number of events or “hits.”
   - **Cost**—total amount of the expense per 100 megabytes.
   - **MBytes**—number of megabytes.
   - **% of MBytes—**percentage of megabytes transferred by this user, compared to all users. For example, if 1000 megabytes of data was transferred during this period and 200 megabytes was transferred by the top user, the **% of MBytes** field will display 20%.
6. The ViewPoint Reporting Module shows yesterday’s report. To change the date range of the report and other settings, click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
8. To display a limited group of users, enter the user IDs in the **Search Bar** fields.



**Note** The search bar fields use pattern matching with operators such as “contains”. For example, “john” will match john_smith, john42, or big_john.

9. When you are finished, click **Search.** The ViewPoint Reporting Module displays the report for the selected users and date range.



**Note** These settings will stay in effect for all similar reports during your active login session.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1433
Keysight, EX-1045
Page 1433

# Viewing Services Reports

Service reports provide information on the amount of data transmitted through the selected SonicWALL appliance by each service.

Service reports are useful for revealing inappropriate usage of bandwidth and can help determine network policies. For example, if there is a large spike of bandwidth usage, you can determine whether this is caused by regular Web access, someone using FTP to transfer large files, an attempted Denial of Service (DoS) attack, or another service.



**Note** All reports appear in the appliance's time zone.

The procedures for viewing the Services Reports are described in the following section:

- "Viewing the Services Summary Report" on page 189



**Note** You cannot view services reports from the global view.

## Viewing the Services Summary Report

The Services Summary report displays the amount of traffic handled by each service during each hour of the specified day.

To view the Services Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1434
Keysight, EX-1045
Page 1434

3. Expand the **Services** tree and click **Summary**. The Summary page displays.




4. The bar graph displays the amount of bandwidth used by each service during each hour of the day.
5. The table contains the following information:
   - **Protocol**—the service.
   - **Events**—number of events or “hits.”
   - **MBytes**—Number of Megabytes.
   - **% of MBytes**—percentage of megabytes transferred by this service on the selected day, compared to all other services. For example, if 10,000 megabytes of data was transferred during the day and 5,000 of the megabytes were transferred, the **% of MBytes** field will display 50%.
6. To change the date of the report and other settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See “Managing Report Settings” on page 154.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1435
Keysight, EX-1045
Page 1435

8. When you are finished, click **Search.** The ViewPoint Reporting Module displays the report for the selected date.

**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Web Usage Reports

Web usage reports provide information on the amount of Web usage that occurs through the selected SonicWALL appliance(s).

Web usage reports can be used to view Web bandwidth usage by the hour, day, or over a period of days. Additionally, you can view the top users of Web bandwidth and view the most visited sites. Exclusion settings for Web sites and Web users are available on the Console > User Settings > Reports page. Web Usage reports will not contain references to the Web sites or users specified on this page. For more information, see the “Configuring Reports Settings” section on page 53.

For the Summary and Over Time reports, and for all reports involving Users, the browse time is also provided in one column of the table. The browse time is the amount of time consumed browsing the Internet through one or more selected SonicWALL appliances. The browse time is not displayed in reports for Category or Sites.

**Note** All reports appear in the appliance’s time zone.

Select from the following:


- “Viewing the Web Usage Summary Report” on page 192
- “Viewing the Top Web Sites” on page 194
- “Viewing the Top Users of Web Bandwidth” on page 195
- “Viewing Web Usage by User” on page 197
- “Viewing Web Usage By Site” on page 199
- “Viewing Web Usage By Category” on page 200
- “Viewing Web Usage Over Time” on page 202
- “Viewing Top Sites Over Time” on page 203
- “Viewing Top Users Over Time” on page 205
- “Viewing Web Usage By User Over Time” on page 207




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1436
Keysight, EX-1045
Page 1436

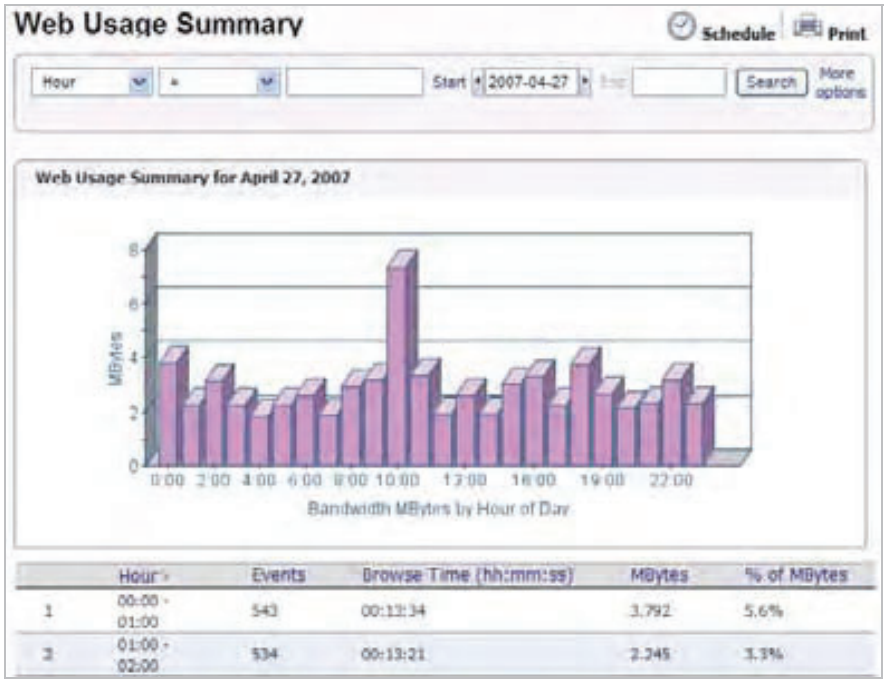
- “Viewing Web Usage By Category Over Time” on page 208


# Viewing the Web Usage Summary Report

The Web Usage Summary report contains information on the amount of HTTP bandwidth handled by a SonicWALL appliance or all SonicWALL appliances during each hour of the specified day. The report includes information on the amount of time spend browsing the Internet behind a SonicWALL appliance or all SonicWALL appliances.

To view the Web Usage Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Web Usage** tree and click **Summary**. The Summary page displays.




4. The bar graph displays the amount of HTTP bandwidth transferred during each hour of the day.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1437
Keysight, EX-1045
Page 1437

5. The table contains the following information:
   - **Hour**—when the sample was taken.
   - **Events**—number of events or “hits.”
   - **Browse Time**—number of hours, minutes, and seconds spent browsing non-job function-related sites on the Internet.

     Browse Time is calculated as follows:

     (Number Of Pages / Noise Reduction Factor) * Average Browse Time Per Page

     "Number Of Pages" is the number of hits (responses by the Web site to build the page) when a User accesses a Web page (www.sonicwall.com).

     "Noise Reduction Factor" is the average noise we want to exclude per page (like eliminating pop-up links, images, and more). The factory default is 40.

     "Average Browse Time Per Page" is the time allocated to read a page.

     Noise Reduction Factor and Average Browse Time Per page are configurable in the database directly, but are not exposed in ViewPoint management interface.
   - **MBytes**—number of megabytes transferred.
   - **% of MBytes**—percentage of megabytes transferred during this hour, compared to the day. For example, if 1000 megabytes of HTTP data was transferred during the day and 100 megabytes was transferred at the 12:00 time period, the **% of MBytes** field will display 10%.
6. To change the date of the report and other settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search.** The ViewPoint Reporting Module displays the report for the selected date.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1438
Keysight, EX-1045
Page 1438

# Viewing the Top Web Sites

The Top Sites report displays the Web sites that used the most HTTP bandwidth on the specified date. To view the Top Sites report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Usage** tree and click **Top Sites**. The Top Sites page displays.




4. The pie chart displays the percentage of bandwidth used to access the top sites.
5. The table contains the following information:
   - **Site**—URL or IP address of the site.
   - **Hits**—number of hits.
   - **MBytes**—number of megabytes transferred.
   - **Category**—the Web site category.
   - **% of MBytes**—percentage of megabytes transferred between this site, compared to all other HTTP traffic. For example, if 10,000 megabytes of data was transferred during the day and 5,000 megabytes was transferred between the appliance and Ebay, the **% of MBytes** field will display 50% and you have a problem.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1439
Keysight, EX-1045
Page 1439

6. To change the date of the report and other settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Sites**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. When you are finished, click **Search.** The ViewPoint Reporting Module displays the report for the selected date.



**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing the Top Users of Web Bandwidth

The Top Users report displays the users who used the most HTTP bandwidth and the amount of time they spent browsing the Internet on the specified date.

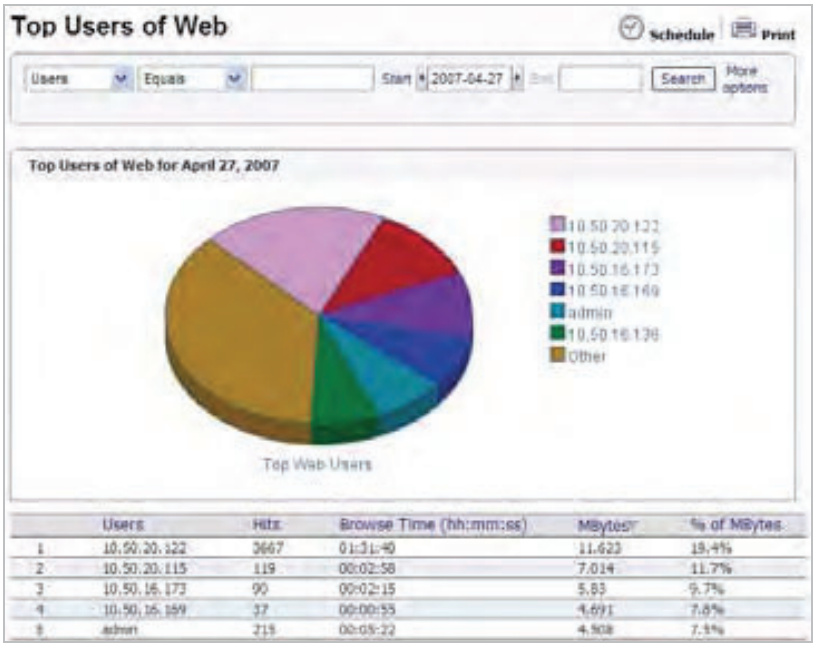
To view the Top Users report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1440
Keysight, EX-1045
Page 1440

3. Expand the **Web Usage** tree and click **Top Users**. The Top Users page displays.




4. The pie chart displays the percentage of bandwidth transferred by each of the top users.
5. The table contains the following information:
   - **Users**—the IP address of the user.
   - **Hits**—number of hits.
   - **Browse Time**—number of hours, minutes, and seconds spent browsing non-job function-related sites on the Internet.
   - **MBytes**—number of megabytes transferred.
   - **% of MBytes**—percentage of megabytes transferred by this user, compared to all users. For example, if 1000 megabytes of data was transferred during the day and 200 megabytes was transferred by the top user, the **% of MBytes** field will display 20%.
6. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report and other settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1441
Keysight, EX-1045
Page 1441

7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. To display a limited group of users, enter the user IDs in the **Search Bar** fields.



**Note** The search bar fields use pattern matching with operators such as "contains". For example, "john" will match john_smith, john42, or big_john.

9. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected day.



**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Web Usage by User

The By User report displays a list of all users, their top sites, the number of hits to each site, the time spent browsing, and the amount of data transferred.

To view the By User report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1442
Keysight, EX-1045
Page 1442

3. Expand the **Web Usage** tree and click **By User**. The By User page displays.




4. The table contains the following information:
   - **User**—the IP address of the user.
   - **Hits**—the number of hits to each Web site visited by the user.
   - **Browse Time**—number of hours, minutes, and seconds spent browsing non-job function-related sites on the Internet.
   - **MBytes**—the number of megabytes transferred.
5. You can navigate directly from the **Web Usage > By User** page to a **Web Usage > By Site** page detailing the information of the site the user has been browsing. Click the **Plus** sign to the left of the **User** name or IP address to show details, and then hover the mouse over a site. A sticky tooltip will display with a link to the corresponding site's report page.
6. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report and other settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Number of Users**
   - **Number of Sites per User**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. To display a limited group of users, enter the user IDs in the **Search Bar** fields.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1443
Keysight, EX-1045
Page 1443

**Note** The search bar fields use pattern matching with operators such as “contains”. For example, “john” will match john_smith, john42, or big_john.

9. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected day.

**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Web Usage By Site

The By Site report displays a list of all sites, the users that accessed the sites, the number of hits to each site, and the amount of data transferred.

To view the By Site report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Usage** tree and click **By Site**. The By Site page displays.




4. The table contains the following information:
   - **Site**—the URL of the site.
   - **Hits**—the number of hits to the Web site, by user.
   - **MBytes**—the number of megabytes transferred, by the user.
   - **Category**—the category of the site.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1444
Keysight, EX-1045
Page 1444

5. You can navigate directly from the **Web Usage > By Site** page to a **Web Usage > By User** page detailing the information of the users who have been browsing the site. Click the **Plus** sign to the left of the **Site** to show details, and then hover the mouse over a user. A sticky tooltip will display with a link to the corresponding user report page.
6. The ViewPoint Reporting Module shows yesterday's report and all Web sites. To change the date of the report or Web sites displayed, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Chart Types you can set:
   - **Number of Sites**
   - **Number of Users per Site**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. To display a limited group of sites, enter the sites in the **Search Bar** fields.



**Note** The search bar fields use pattern matching with operators such as "contains". For example, "john" will match john_smith, john42, or big_john.

9. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected day.



**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Web Usage By Category

The Web Usage By Category report displays a list of the top Web site categories, the number of hits to each category, the amount of data transferred, and the percentage of data transferred.

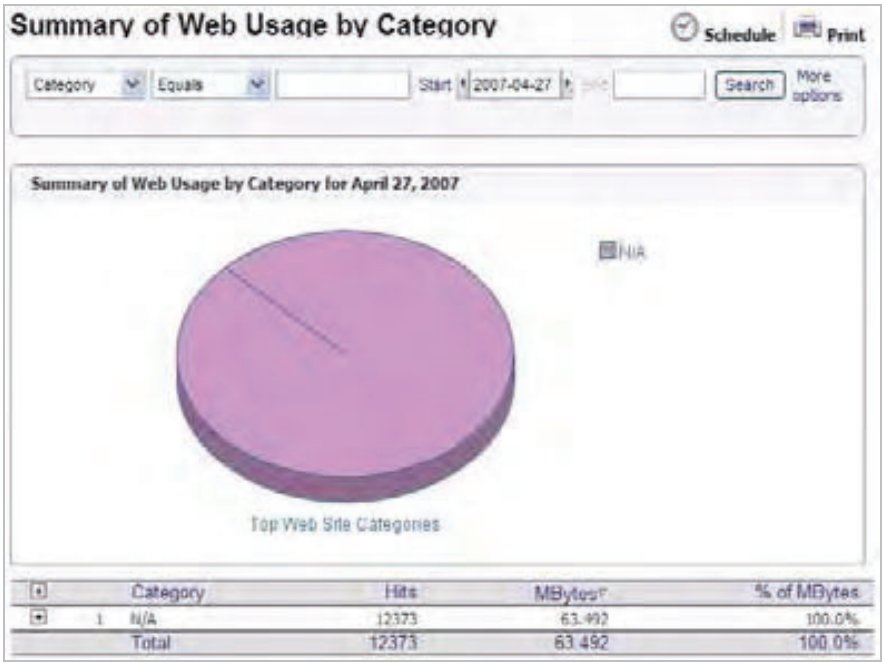
To view the By Category report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1445
Keysight, EX-1045
Page 1445

3. Expand the **Web Usage** tree and click **By Category**. The By Category page displays.




4. The table contains the following information:
   - **Category**—the Web site category.
   - **Hits**—the number of hits to the Web site category.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred.
5. The ViewPoint Reporting Module shows yesterday's report and all Web site categories. To change the date of the report or Web site categories displayed, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
7. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected day.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1446
Keysight, EX-1045
Page 1446

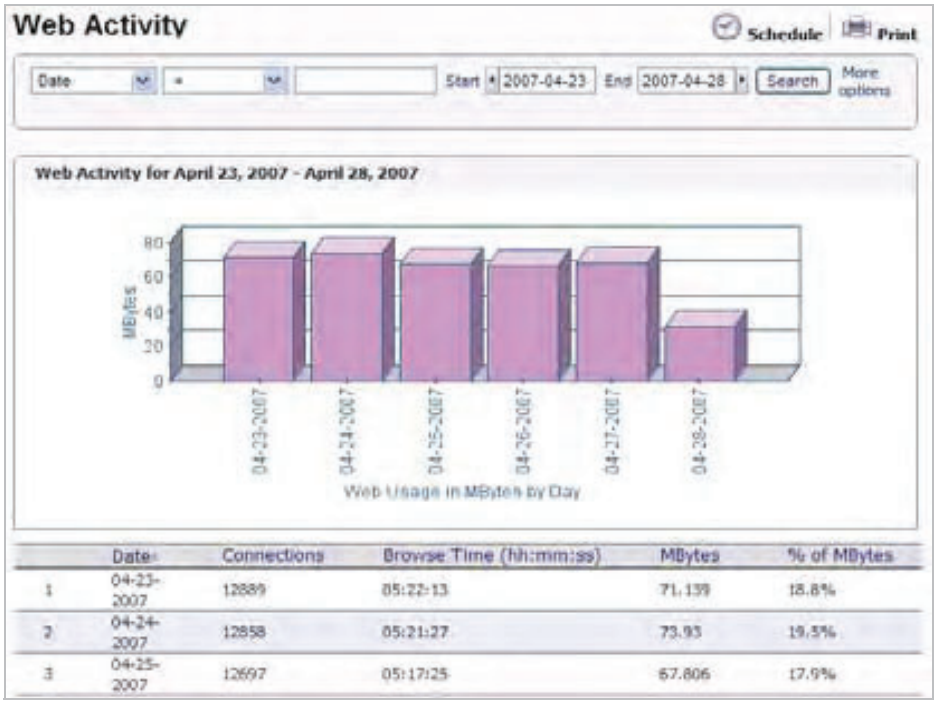
**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Web Usage Over Time

The Web Usage Over Time report displays the daily amount of HTTP bandwidth and browse time handled by a SonicWALL appliance or all SonicWALL appliances for the specified time period.

To view the Web Usage Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Web Usage** tree and click **Over Time**. The Web Activity page displays.




4. The bar graph displays the amount of HTTP bandwidth transferred during each day of the specified time period.
5. The table contains the following information:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1447
Keysight, EX-1045
Page 1447

- **Date**—when the sample was taken.
- **Connections**—the number of connections or hits.
- **Browse Time**—number of hours, minutes, and seconds spent browsing non-job function-related sites on the Internet.
- **MBytes**—the number of megabytes transferred.
- **% of MBytes**—the percentage of megabytes transferred during this day, compared to the time period. For example, if 100,000 megabytes of data was transferred during the time period and 25,000 megabytes was transferred on one day, the **% of MBytes** field will display 25%.

6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Top Sites Over Time

The Top Sites Over Time report displays the most visited Web sites for the specified time period.

To view the Top Sites Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1448
Keysight, EX-1045
Page 1448

3. Expand the **Web Usage** tree and click **Top Sites Over Time**. The Top Sites Over Time page displays.




4. The bar graph displays the amount of HTTP bandwidth transferred during each day of the specified time period.
5. The table contains the following information:
   - **Site**—URL or IP address of the site.
   - **Hits**—the number of hits.
   - **MBytes**—the number of megabytes transferred.
   - **Category**—the Web site category.
   - **% of MBytes**—the percentage of megabytes transferred between this site, compared to all other HTTP traffic. For example, if 1,000,000 megabytes of data was transferred during the day and 500,000 megabytes was transferred between the appliance and Ebay, the **% of MBytes** field will display 50% and you have a problem.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Sites**



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1449
Keysight, EX-1045
Page 1449

– **Rows per Screen**

See “Managing Report Settings” on page 154.

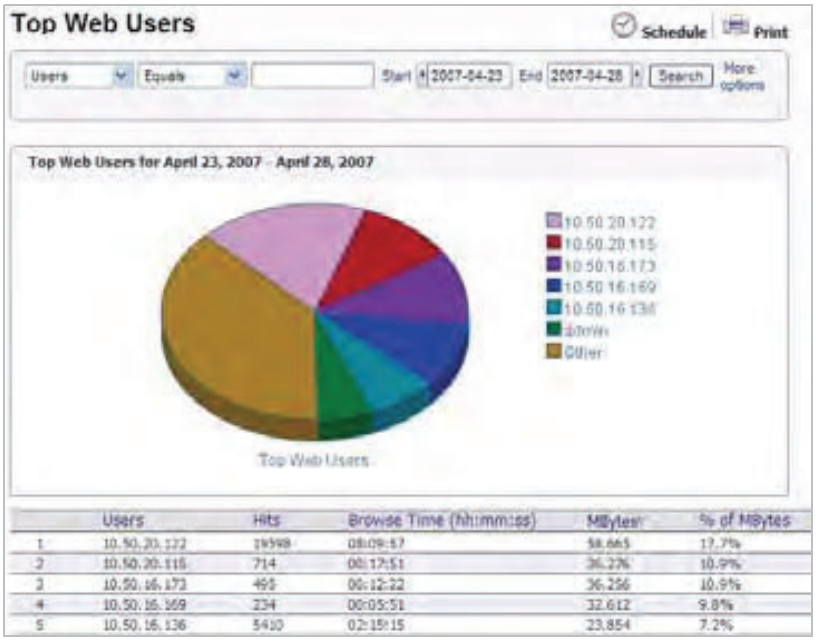
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Top Users Over Time

The Top Users Over Time report displays the top users of bandwidth and the amount of time they spent browsing the Internet for the specified time period. To view the Top Users Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Usage** tree and click **Top Users Over Time**. The Top Users Over Time page displays.




4. The graph provides a graphical display of the percentage of bandwidth transferred by each of the top users over the specified time period.
5. The table contains the following information:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1450
Keysight, EX-1045
Page 1450

- **Site**—URL or IP address of the site.
- **Hits**—number of hits.
- **Browse Time**—number of hours, minutes, and seconds spent browsing non-job function-related sites on the Internet.
- **MBytes**—number of megabytes transferred.
- **Category**—the category of the site.
- **% of MBytes**—percentage of megabytes transferred by this user, compared to all users. For example, if 1000 megabytes of data was transferred during the period and 200 megabytes was transferred by the top user, the **% of MBytes** field will display 20%.

6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



**Note** These settings will stay in effect for all similar reports during your active login session.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1451
Keysight, EX-1045
Page 1451

# Viewing Web Usage By User Over Time

The By User Over Time report displays a list of all users, their top sites, the number of hits to each site, the time spent browsing, and the amount of data transferred for the specified time period.

To view the By User Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Usage** tree and click **By User Over Time**. The By User Over Time page displays.




4. The table contains the following information:
   - **User**—the IP address of the user.
   - **Hits**—number of hits to each Web site visited by the user.
   - **Browse Time**—number of hours, minutes, and seconds spent browsing non-job function-related sites on the Internet.
   - **MBytes**—number of megabytes transferred.
5. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
6. Under Report Display Settings you can set:
   - **Number of Users**
   - **Number of Sites per User**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
7. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1452
Keysight, EX-1045
Page 1452

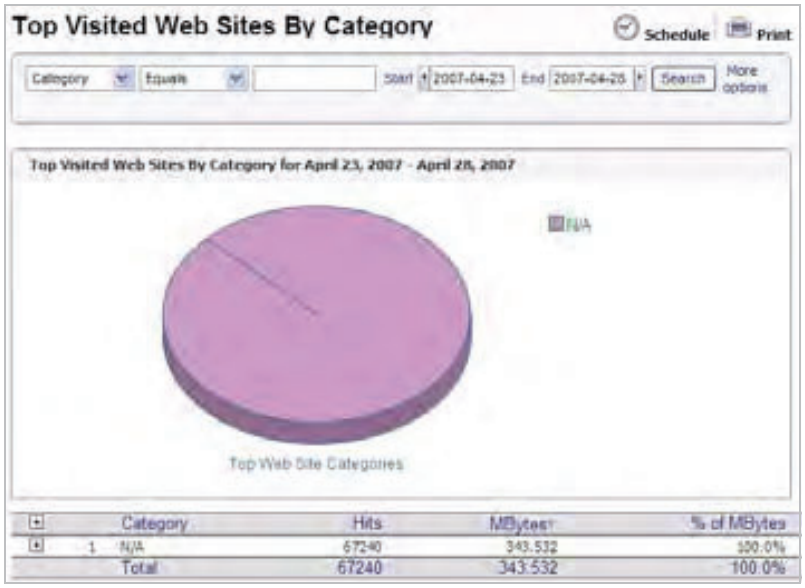
**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Web Usage By Category Over Time

The By Category Over Time report displays a list of all users, their top sites, the number of hits to each site, and the amount of data transferred for the specified time period.

To view the By Category Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Usage** tree and click **By Category Over Time**. The By User Over Time page displays.




4. The table contains the following information:
   - **Category**—the Web site category.
   - **Hits**—number of hits to each Web site visited by the user.
   - **MBytes**—number of megabytes transferred.
   - **% of MBytes**—percentage of megabytes transferred by this user, compared to all users. For example, if 1000 megabytes of data was transferred during the period and 200 megabytes was transferred by the top user, the **% of MBytes** field will display 20%.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1453
Keysight, EX-1045
Page 1453

5. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
7. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Web Filter Reports

Web filter reports provide information on the number of attempts that users made to access blocked Web sites through the selected SonicWALL appliance(s). These reports include Web sites blocked by the Content Filter List, customized keyword filtering, and domain name filtering.

Web filter reports can be used to view blocked site access attempts by the hour, day, or over a period of days. Additionally, you can view the users that most frequently attempt to access blocked sites and the most popular blocked sites.



**Note** All reports appear in the applianceUTM’s time zone.

Select from the following:


- “Viewing the Web Filter Summary Report” on page 210
- “Viewing the Web Filter Top Sites Report” on page 212
- “Viewing the Top Users that Try to Access Blocked Sites” on page 213
- “Viewing the Blocked Sites for Each User” on page 215
- “Viewing Blocked Sites Sorted By Site” on page 216




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1454
Keysight, EX-1045
Page 1454

- “Viewing Blocked Sites Sorted By Category” on page 217
- “Viewing Blocked Site Attempts Over Time” on page 219
- “Viewing the Top Blocked Site Attempts Over Time” on page 220
- “Viewing the Top Blocked Site Users Over Time” on page 221
- “Viewing Blocked Sites for Each User Over Time” on page 222
- “Viewing Blocked Sites By Category Over Time” on page 223


# Viewing the Web Filter Summary Report

The Web Filter Summary report contains information on the number of times users attempt to access blocked sites for the specified day.

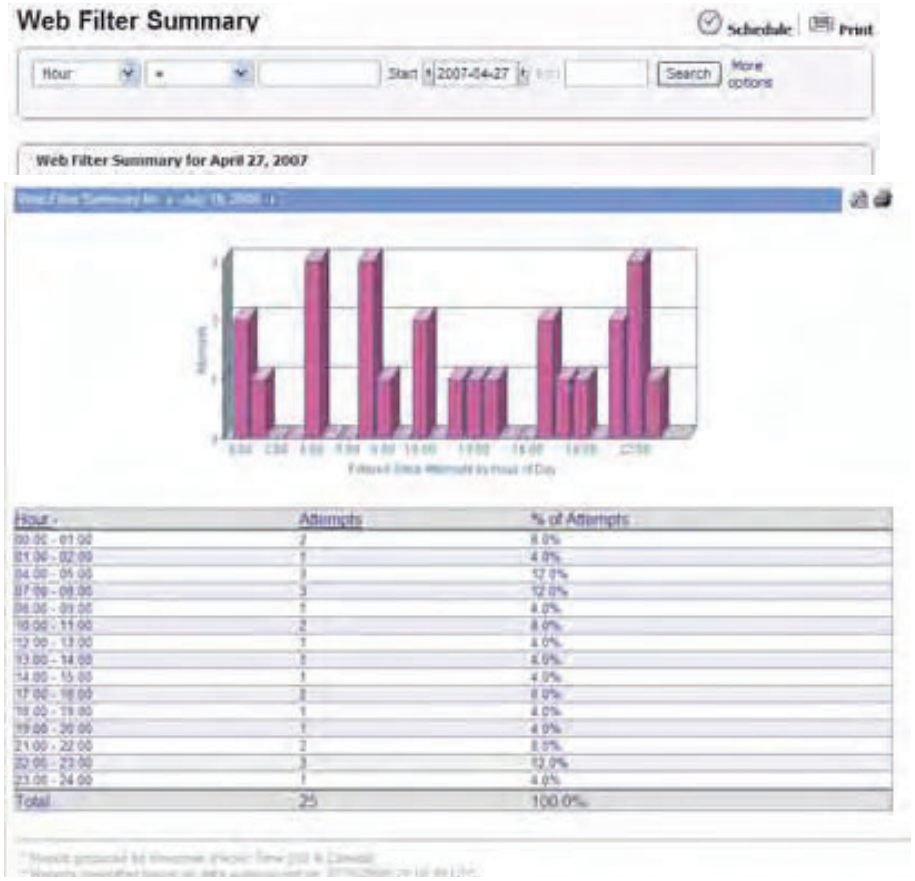
To view the Web Filter Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1455
Keysight, EX-1045
Page 1455

3. Expand the **Web Filter** tree and click **Summary**. The Summary page displays.




4. The bar graph displays the number of blocked sites that users attempted to access during each hour of the day.
5. The table contains the following information:
   - **Hour**—time when the sample was taken.
   - **Attempts**—the number of attempts to access blocked sites.
   - **% of Attempts**—the percentage of attempts during this hour, compared to the day. For example, if 100 attempts occurred during the day and 20 attempts occurred at the 12:00 time period, the **% of Attempts** field will display 20%.
6. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1456
Keysight, EX-1045
Page 1456

7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

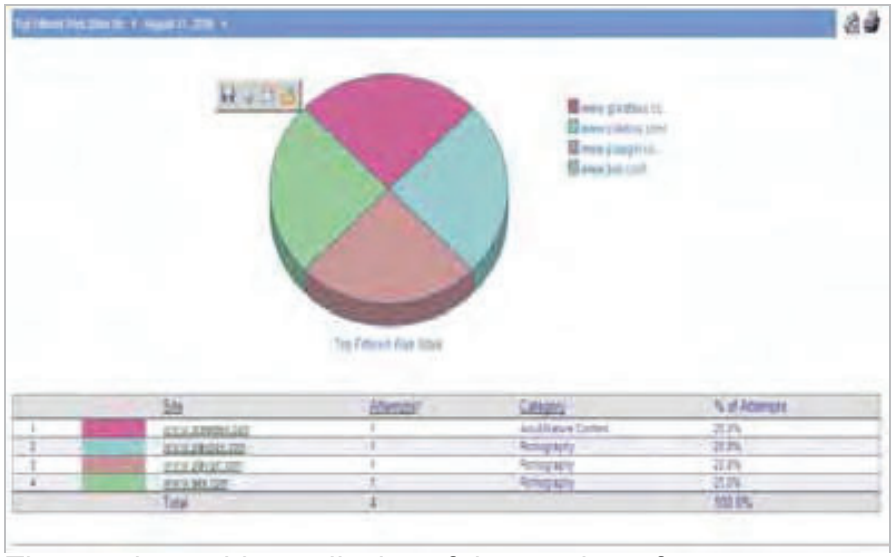
   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing the Web Filter Top Sites Report

The Web Filter Top Sites report displays the top blocked Web sites that users attempted to access on the specified date.

To view the Top Sites report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Filter** tree and click **Top Sites**. The Top Sites page displays.



4. The graph provides a display of the number of access attempts for each of the top twenty blocked Web sites.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1457
Keysight, EX-1045
Page 1457

5. The table contains the following information:
   - **Site**—the URL or IP address of the site.
   - **Attempts**—the number of attempts.
   - **Category**—the Web site category.
   - **% of Attempts**—percentage of attempts to access the blocked site, compared to all other blocked site attempts. For example, if 500 attempts were made during the day and 100 of those attempts were for www.badsite.com, its **% of Attempts** field will display 20%.
6. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Sites**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing the Top Users that Try to Access Blocked Sites

The Web Filter Top Users report displays the users who made the most attempts to access blocked sites on the specified date.

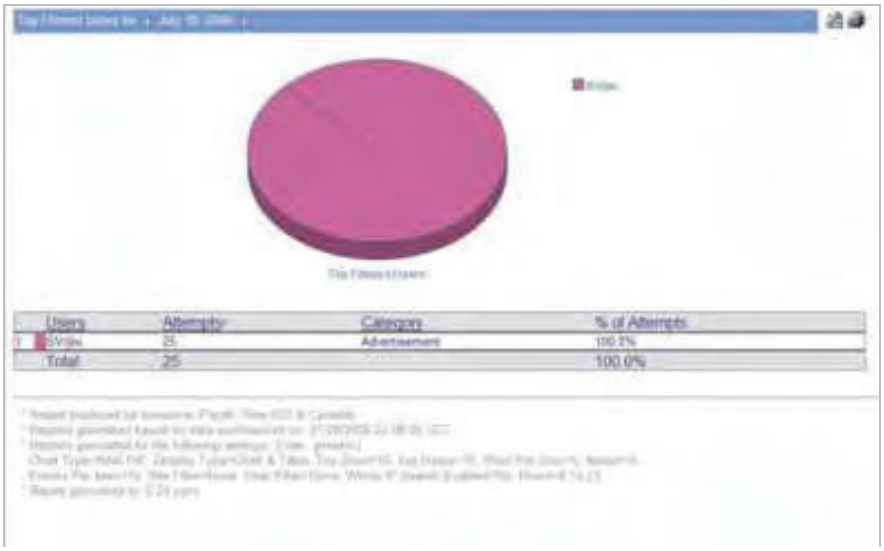
To view the Top Users report, perform the following steps:

1. Click the **UTM** tab.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1458
Keysight, EX-1045
Page 1458

2. Select a SonicWALL appliance.
3. Expand the **Web Filter** tree and click **Top Users**. The Top Users page displays.



4. The pie chart displays the top users with the most blocked site attempts.
5. The table contains the following information:
   - **Users**—the IP address of the user.
   - **Attempts**—the number of attempts.
   - **Category**—the Web site category.
   - **% of Attempts**—percentage of attempts to access the blocked site, compared to all other user attempts. For example, if 500 attempts were made during the day and 250 of those attempts were made by a single user, that user's **% of Attempts** field will display 50%.
6. By default, ViewPoint Reporting shows yesterday's report, a pie chart, and the ten top users. To change these settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table** or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1459
Keysight, EX-1045
Page 1459

8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.
9. These settings will stay in effect for all similar reports during your active login session.

# Viewing the Blocked Sites for Each User

The Web Filter By User report displays the top blocked Web sites that each user attempted to access on the specified date.

To view the Web Filter By User report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Filter** tree and click **By User**. The By User page displays.



4. The table contains the following information:
   - **User**—the IP address of the user.
   - **Site**—the top five sites visited by the user.
   - **Attempts**—the number of attempts the user made to access each Web site.
5. You can navigate directly from the **Web Filter > By User** page to a **Web Filter > By Site** page detailing the information of the site the user has been browsing. Click the **Plus** sign to the left of the **User** name or IP address to show details, and then hover the mouse over a site. A sticky tooltip will display with a link to the corresponding site's report page.
6. By default, the ViewPoint Reporting Module shows yesterday's report, a pie chart, and the ten top users. To change these settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1460
Keysight, EX-1045
Page 1460

- **Number of Users**
- **Number of Sites per User**
- **Rows per Screen**

See “Managing Report Settings” on page 154.

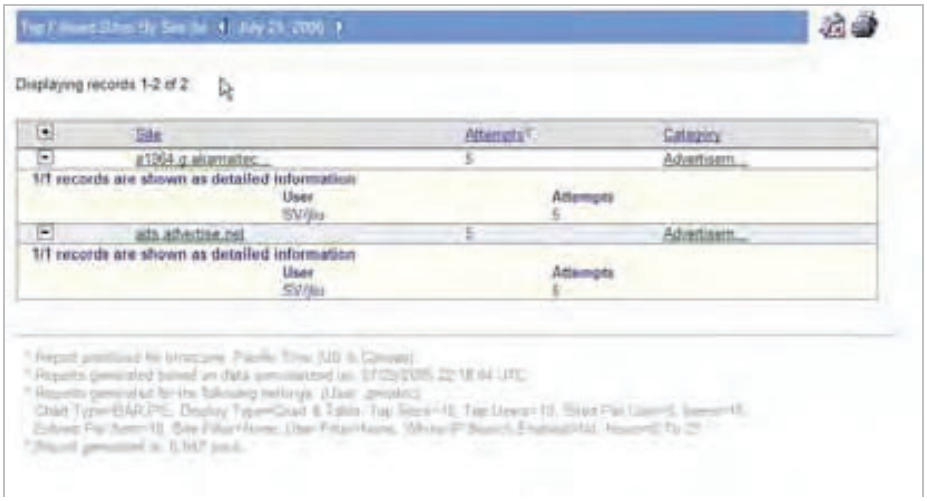
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected settings.
9. These settings will stay in effect for all similar reports during your active login session.

# Viewing Blocked Sites Sorted By Site

The Web Filter By Site report displays the top blocked Web sites that were accessed by users.

To view the Web Filter By Site report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Filter** tree and click **By Site**. The By Site page displays.




4. The table contains the following information:
   - **Site**—the top five sites visited by the user.
   - **Attempts**—the number of attempts the user made to access each Web site.
   - **Category**—the Web site category.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1461
Keysight, EX-1045
Page 1461

5. You can navigate directly from the **Web Filter > By Site** page to a **Web Filter > By User** page detailing the information of the users who have been browsing the site. Click the Plus sign to the left of the Site to show details, and then hover the mouse over a user. A sticky tooltip will display with a link to the corresponding user report page.
6. By default, the ViewPoint Reporting Module shows yesterday's report, a pie chart, and the ten top users. To change these settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Number of Users per Site:
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. Search for Web site addresses in the Search Bar fields.
9. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing Blocked Sites Sorted By Category

The Web Filter By Category report displays the top categories of Web sites that were accessed by users.

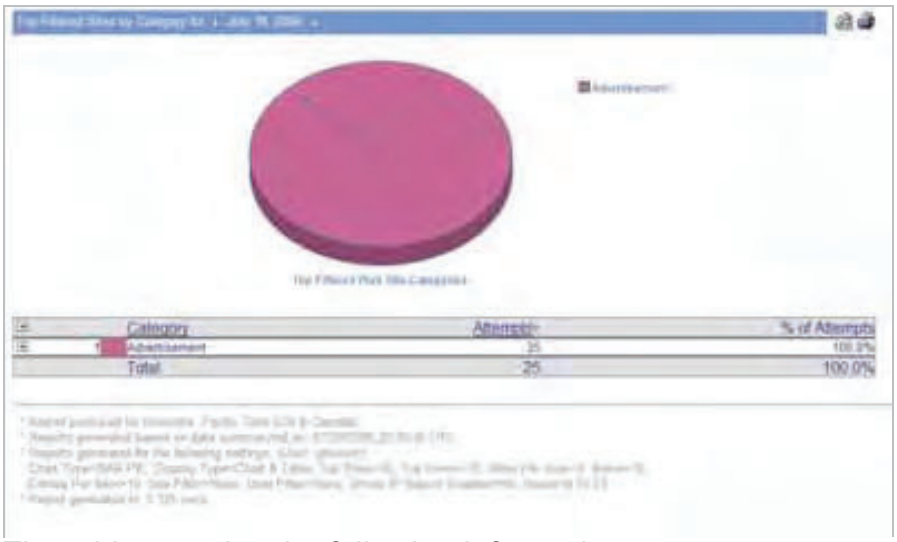
To view the Web Filter By Category report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1462
Keysight, EX-1045
Page 1462

3. Expand the **Web Filter** tree and click **By Category**. The By Site page displays.



4. The table contains the following information:
   - **Category**—the Web site category.
   - **Attempts**—the number of attempts the user made to access each Web site.
   - **% of Attempts**—the percentage of attempts to access the blocked site, compared to all other user attempts. For example, if 500 attempts were made during the day and 250 of those attempts were made by a single user, his **% of Attempts** field will display 50%.
5. By default, the ViewPoint Reporting Module shows yesterday's report, a pie chart, and the ten top users. To change these settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
7. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



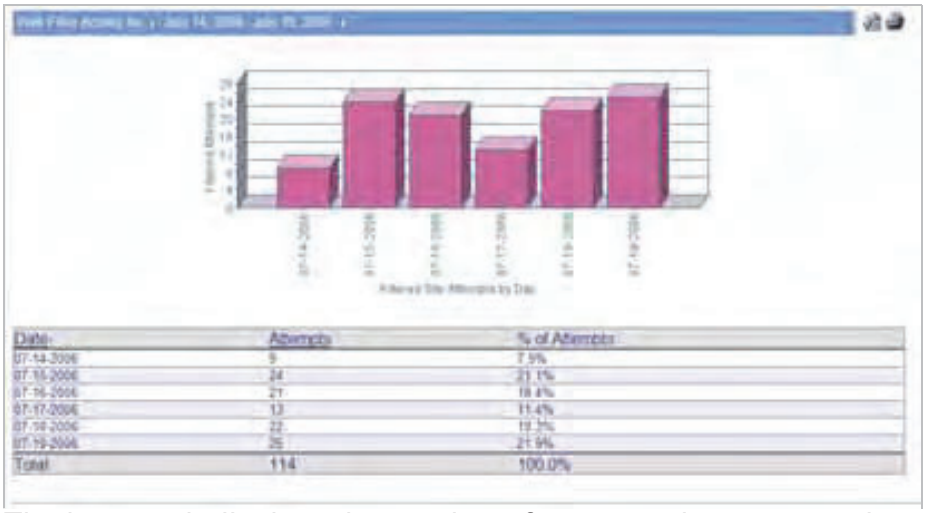
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1463
Keysight, EX-1045
Page 1463

# Viewing Blocked Site Attempts Over Time

The Web Filter Over Time report displays the number of attempts that were made to access blocked Web sites for the specified time period.

To view the Web Filter Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Web Filter** tree and click **Over Time**. The Over Time page displays.



| Date | Attempts | % of Attempts |
|---|---|---|
| 07-14-2006 | 9 | 7.9% |
| 07-15-2006 | 24 | 21.1% |
| 07-16-2006 | 21 | 18.4% |
| 07-17-2006 | 13 | 11.4% |
| 07-18-2006 | 22 | 19.3% |
| 07-19-2006 | 25 | 21.9% |
| Total | 114 | 100.0% |

4. The bar graph displays the number of attempts that were made to access blocked Web sites during each day of the specified time period.
5. The table contains the following information:
   - **Date**—the day when the sample was taken.
   - **Attempts**—the number of attempts to access blocked Web sites.
   - **% of Attempts**—the percentage of attempts to access the blocked site on the day, compared to the time period. For example, if 5,000 attempts were made during the time period and 500 were made on one day, its **% of Attempts** field will display 10%.
6. To change date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1464
Keysight, EX-1045
Page 1464

See “Managing Report Settings” on page 154.

8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

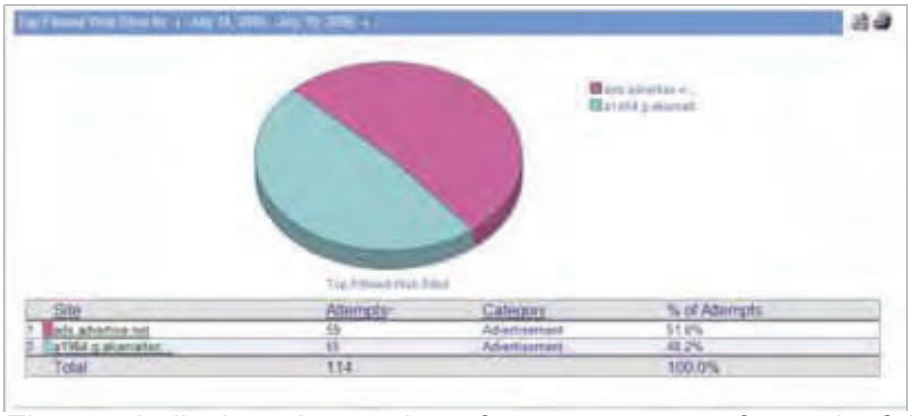
**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing the Top Blocked Site Attempts Over Time

The Top Sites Over Time report displays the top blocked Web sites for the specified time period.

To view the Web Filter Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Filter** tree and click **Top Sites Over Time**. The Top Sites Over Time page displays.




4. The graph displays the number of access attempts for each of the top blocked Web sites during the specified time period.
5. The table contains the following information:
   - **Site**—the URL or IP address of the site.
   - **Attempts**—the number of attempts.
   - **Category**—the Web site category.
   - **% of Attempts**—the percentage of attempts to access the blocked site, compared to all other blocked site attempts. For example, if 500 attempts were made during the period and 100 of those attempts were for www.badsite.com, its **% of Attempts** field will display 20%.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1465
Keysight, EX-1045
Page 1465

6. To change date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Sites**
   - **Rows per Screen**

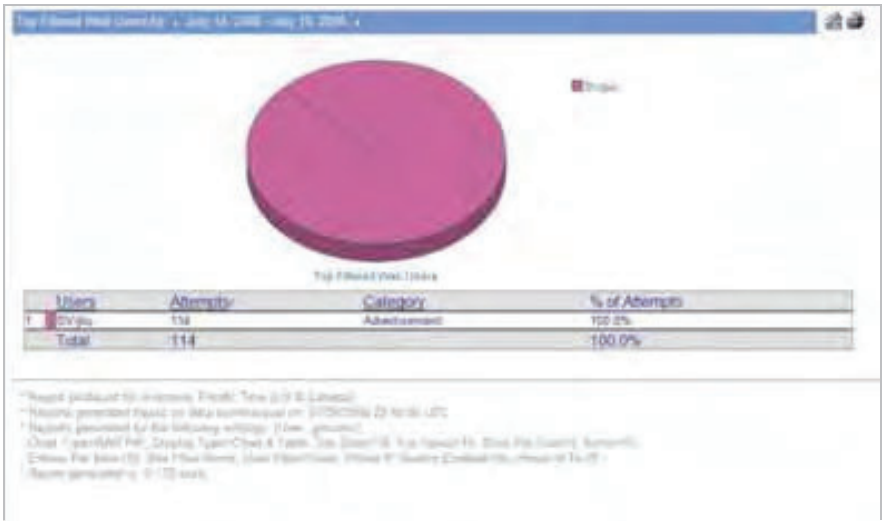
   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing the Top Blocked Site Users Over Time

The Web Filter Top Users Over Time report displays the users who made the most attempts to access blocked sites during the specified time period.

To view the Top Users Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Filter** tree and click **Top Users Over Time**. The Top Users Over Time page displays.




4. The pie chart displays the top users with the most blocked site attempts.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1466
Keysight, EX-1045
Page 1466

5. The table contains the following information:
   - **Users**—the IP address of the user.
   - **Attempts**—the number of attempts.
   - **Category**—the Web site category.
   - **% of Attempts**—the percentage of attempts to access the blocked site, compared to all other user attempts. For example, if 500 attempts were made during the period and 250 of those attempts were made by a single user, his **% of Attempts** field will display 50%.
6. To change date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Sites**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing Blocked Sites for Each User Over Time

The Web Filter By User report displays the top blocked Web sites that each user attempted to access during the specified time period.

To view the By User Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1467
Keysight, EX-1045
Page 1467

3. Expand the **Web Filter** tree and click **By User Over Time**. The By User Over Time page displays.



4. The table contains the following information:
   - **User**—the IP address or name of the user.
   - **Attempts**—the number of attempts the user made to access each Web site.
5. To change date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
7. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

**Note** These settings will stay in effect for all similar reports during your active login session.

# Viewing Blocked Sites By Category Over Time

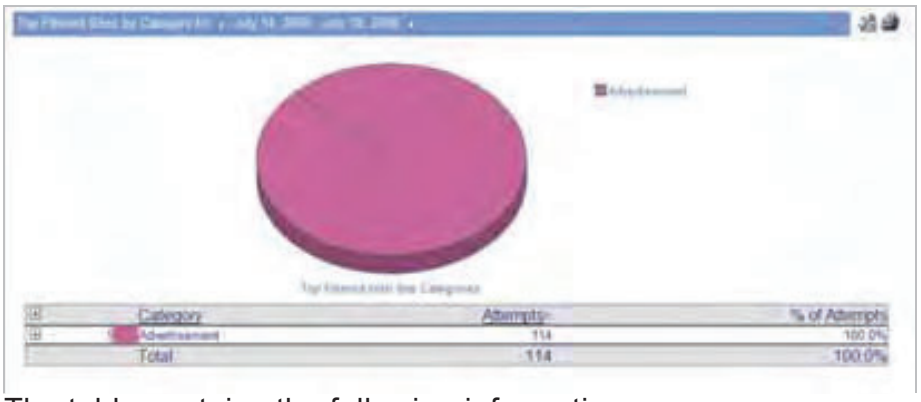
The Web Filter By Category Over Time report displays the top categories that users attempted to access.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1468
Keysight, EX-1045
Page 1468

To view the By Category Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Web Filter** tree and click **By Category Over Time**. The By Category Over Time page displays.




4. The table contains the following information:
   - **Category**—the Web site category.
   - **Attempts**—number of attempts the user made to access each Web site.
   - **% of Attempts**—the percentage of attempts to access the blocked site, compared to all other user attempts. For example, if 500 attempts were made during the period and 250 of those attempts were made by a single user, his **% of Attempts** field will display 50%.
5. To change date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
7. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1469
Keysight, EX-1045
Page 1469

# Viewing File Transfer Protocol Reports

FTP usage reports provide information on the amount of FTP usage that occurs through the selected SonicWALL appliance(s).

FTP usage reports can be used to view FTP bandwidth usage by the hour, day, or over a period of days. Additionally, you can view the top users of FTP bandwidth.

General bandwidth reports do not always provide a complete picture of network bandwidth usage. If a large amount of FTP traffic occurs during peak times, you might need more bandwidth, you might need to upgrade network equipment, or you might ask employees to use compression or transfer large files during non-peak times.

**Note** All reports appear in the appliance's time zone.

Select from the following:


- "Viewing the FTP Summary Report" on page 225
- "Viewing the Top FTP Sites By User" on page 227
- "Viewing FTP Bandwidth Usage Over Time" on page 228
- "Viewing the Top Users of FTP Bandwidth Over Time" on page 230


## Viewing the FTP Summary Report

The FTP Summary report contains information on the amount of FTP bandwidth handled by a SonicWALL appliance or all SonicWALL appliances during the specified day.

To view the FTP Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1470
Keysight, EX-1045
Page 1470

3. Expand the **FTP Usage** tree and click **Summary**. The Summary page displays.




4. The bar graph displays the amount of FTP bandwidth transferred during each hour of the day.
5. The table contains the following information:
   - **Hour**—when the sample was taken.
   - **Events**—the number of FTP events.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred during this hour, compared to the day. For example, if 1000 megabytes of FTP data was transferred during the day and 100 megabytes was transferred at the 12:00 time period, the **% of MBytes** field will display 10%.
6. The ViewPoint Reporting Module shows yesterday's report. To change the date or other report settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart

   See "Managing Report Settings" on page 154.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1471
Keysight, EX-1045
Page 1471

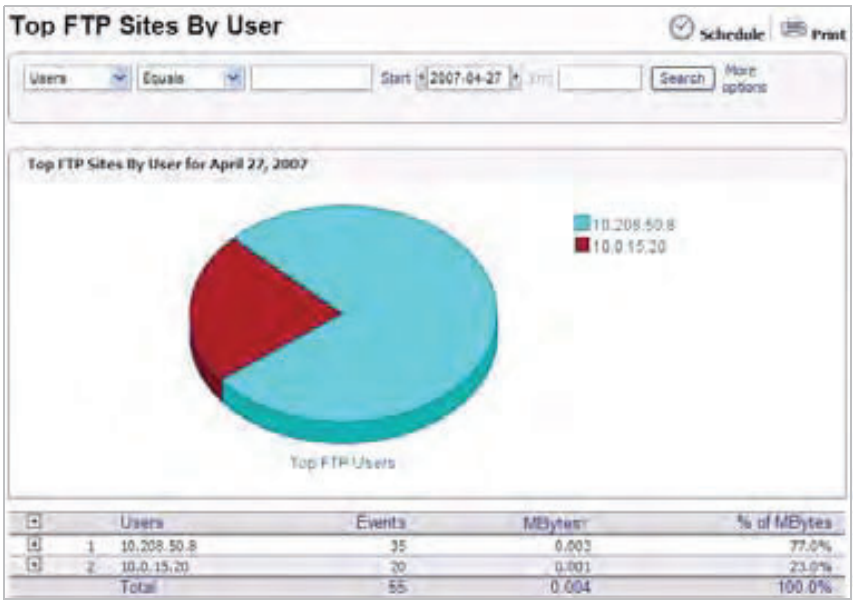
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing the Top FTP Sites By User

The By User report displays the users who used the most FTP bandwidth on the specified date.

To view the By User report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **FTP Usage** tree and click **By User**. The By User page displays.




4. The pie chart displays the percentage of bandwidth used by each user. To view the sites visited by each user, expand the user's site tree (indicated by a '+' sign).
5. The table contains the following information:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1472
Keysight, EX-1045
Page 1472

- **Users**—the IP address of the user.
- **Events**—the number of FTP Events.
- **MBytes**—the number of megabytes transferred.
- **% of MBytes**—the percentage of megabytes transferred during this hour, compared to the day. For example, if 1000 megabytes of FTP data was transferred during the day and 100 megabytes was transferred at the 12:00 time period, the **% of MBytes** field will display 10%.

6. By default, the ViewPoint Reporting Module shows yesterday's report, a pie chart, and the ten top users. To change these settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Number of Sites per User**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. To display a limited group of users, use the Search Bar fields.



**Note** The search bar fields use pattern matching with operators such as "contains". For example, "john" will match john_smith, john42, or big_john.

9. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing FTP Bandwidth Usage Over Time

The FTP Usage Over Time report displays the daily amount of FTP bandwidth handled by a SonicWALL appliance or all SonicWALL appliances for the specified time period.

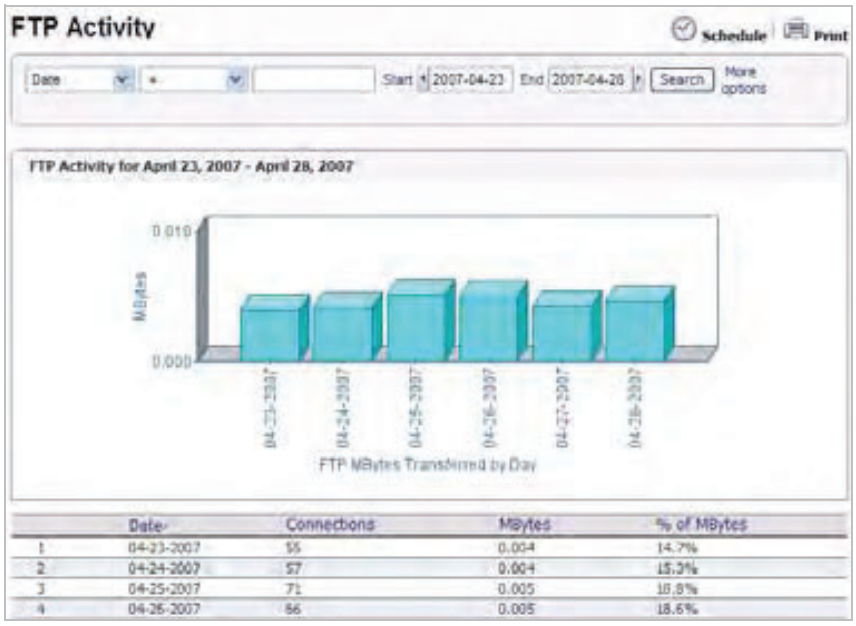
To view the FTP Usage Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1473
Keysight, EX-1045
Page 1473

3. Expand the **FTP Usage** tree and click **Over Time**. The FTP Activity page displays.




4. The bar graph displays the amount of FTP bandwidth transferred during each day of the specified time period.
5. The table contains the following information:
   - **Date**—when the sample was taken.
   - **Connections**—the number of FTP connections.
   - **MBytes**—the number of megabytes transferred.
   - **% of Usage**—the percentage of megabytes transferred during this day, compared to the time period. For example, if 10,000 megabytes of FTP data was transferred during the time period and 2,500 megabytes of FTP data was transferred on one day, the **% of Usage** field will display 25%.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



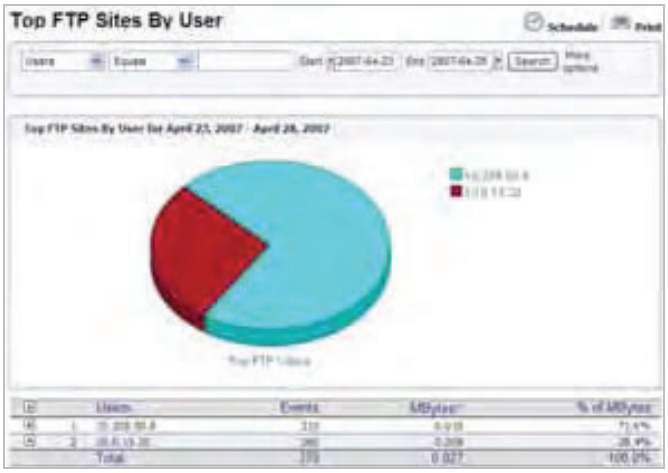
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1474
Keysight, EX-1045
Page 1474

# Viewing the Top Users of FTP Bandwidth Over Time

The By Users Over Time report displays the users who used the most FTP bandwidth for the specified time period.

To view the By Users Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **FTP Usage** tree and click **By Users Over Time**. The By Users Over Time page displays.




4. The table contains the following information:
   - **Users**—the IP address of the user.
   - **Events**—the number of FTP Events.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred by this user, compared to all users. For example, if 10000 megabytes of data was transferred during the period and 2000 megabytes was transferred by the top user, the **% of MBytes** field will display 20%.
5. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1475
Keysight, EX-1045
Page 1475

6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Number of Sites per User**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
7. To display a limited group of users, use the Search Bar fields.



**Note** The search bar fields use pattern matching with operators such as “contains”. For example, “john” will match john_smith, john42, or big_john.

8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing Mail Usage Reports

Mail usage reports provide information on the amount of mail usage that occurs through the selected SonicWALL appliance(s).

Mail usage reports can be used to view mail bandwidth usage by the hour, day, or over a period of days. Additionally, you can view the top users of mail bandwidth.



**Note** Mail usage reports include SMTP, POP3, and IMAP traffic.

General bandwidth reports do not always provide a complete picture of network bandwidth usage. If a large amount of mail traffic occurs during peak times, you might want to take some of the following actions:

- Add bandwidth
- Upgrade network equipment
- Ask employees to use compression or transfer large files during non-peak times
- Ask employees to place large files on an FTP site rather than sending them as mail attachments.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1476
Keysight, EX-1045
Page 1476



**Note** All reports appear in the appliance's time zone.

Select from the following:

- To view a summary of the daily mail usage, see "Viewing the Mail Usage Summary Report" on page 232.
- To view the users who consume the most mail bandwidth, see "Viewing the Top Users of Mail Bandwidth" on page 234.
- To view mail usage over a period of time, see "Viewing Mail Usage Over Time" on page 235.
- To view the users who consume the most mail bandwidth over time, see "Viewing the Top Users of Mail Bandwidth Over Time" on page 237.

# Viewing the Mail Usage Summary Report

The Mail Usage Summary report contains information on the amount of mail handled by a SonicWALL appliance or all SonicWALL appliances during the specified day.

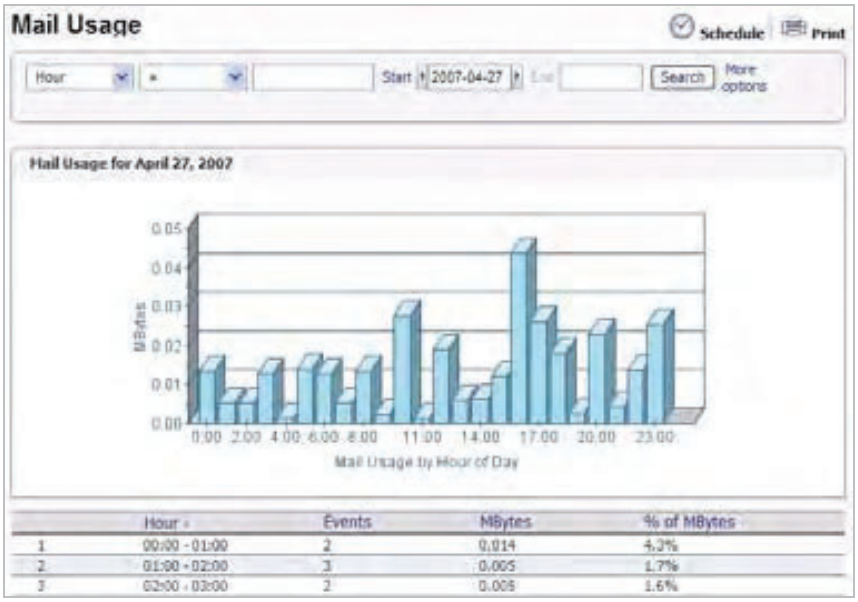
To view the Mail Usage Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1477
Keysight, EX-1045
Page 1477

3. Expand the **Mail Usage** tree and click **Summary**. The Summary page displays.




4. The bar graph displays the amount of mail sent and received during each hour of the day.
5. The table contains the following information:
   - **Hour**—when the sample was taken.
   - **Events**—the number of mail events.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred during this hour, compared to the day. For example, if 10,000 megabytes of mail was transferred during the day and 1,000 megabytes was transferred at the 12:00 time period, the **% of MBytes** field will display 10%.
6. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report or the report display settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See "Managing Report Settings" on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.



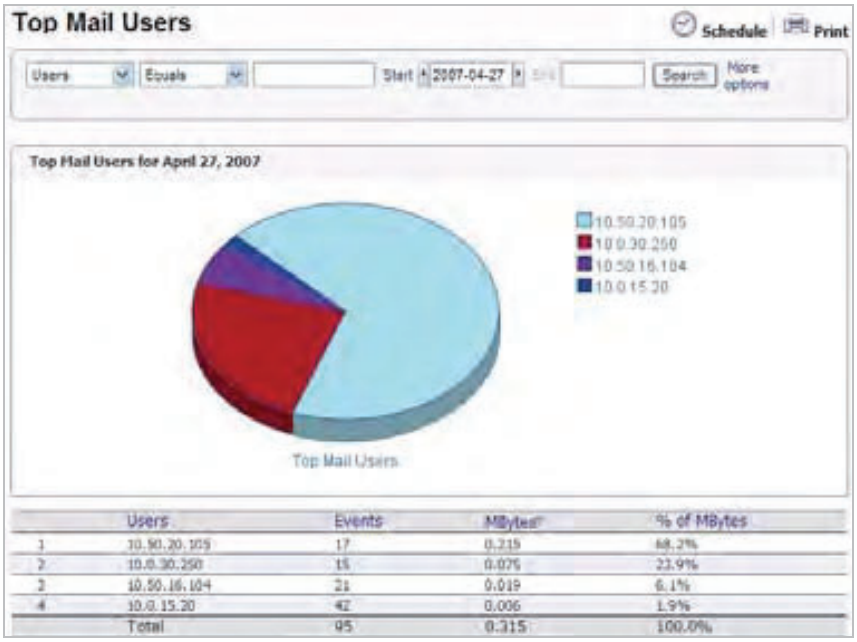
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1478
Keysight, EX-1045
Page 1478

# Viewing the Top Users of Mail Bandwidth

The Top Users report displays the users who sent and received the most mail on the specified date.

To view the Top Users report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Mail Usage** tree and click **Top Users**. The Top Users page displays.




4. The pie chart displays the percentage of mail sent and received by the top mail users.
5. The table contains the following information:
   - **Users**—the IP address of the user.
   - **Events**—the number of mail messages sent and received.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred by this user, compared to all users. For example, if 10000 megabytes of data was transferred during the day and 2000 megabytes was transferred by the top user, the **% of MBytes** field will display 20%.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1479
Keysight, EX-1045
Page 1479

6. By default, the ViewPoint Reporting Module shows yesterday's report, a pie chart, and the ten top users. To change the date of the report or the report display settings, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing Mail Usage Over Time

The Mail Usage Over Time report displays the daily amount of mail handled by a SonicWALL appliance or all SonicWALL appliances for the specified time period.

To view the Mail Usage Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1480
Keysight, EX-1045
Page 1480

3. Expand the **Mail Usage** tree and click **Over Time**. The Over Time page displays.




4. The bar graph displays the amount of mail sent and received during each day of the specified time period.
5. The table contains the following information:
   - **Date**—when the sample was taken.
   - **Connections**—the number of mail messages.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred by this user, compared to all users. For example, if 10000 megabytes of data was transferred during the day and 2000 megabytes was transferred by the top user, the **% of MBytes** field will display 20%.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type: Area, Bar** or **Plot chart**

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



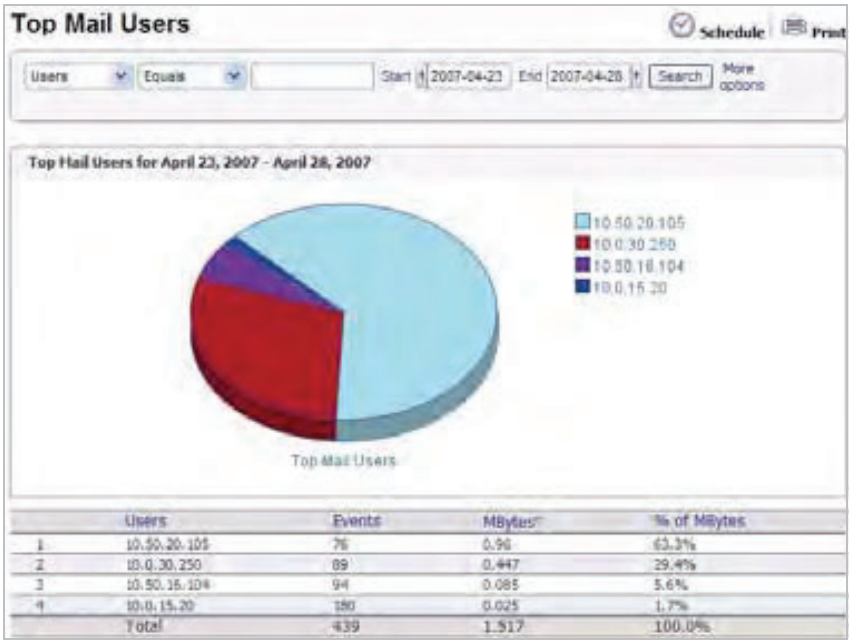
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1481
Keysight, EX-1045
Page 1481

# Viewing the Top Users of Mail Bandwidth Over Time

The Top Users Over Time report displays the users who sent and received the most mail during the specified time period.

To view the Top Users Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Mail Usage** tree and click **Top Users Over Time**. The Top Users Over Time page displays.




4. The pie chart displays the percentage of mail sent and received by the top mail users.
5. The table contains the following information:
   - **Users**—the IP address of the user.
   - **Events**—the number of mail messages sent and received.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred by this user, compared to all users. For example, if 10,000 megabytes of data was transferred during the period and 2000 kilobytes was transferred by the top user, the **% of MBytes** field will display 20%.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1482
Keysight, EX-1045
Page 1482

6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. To display a limited group of users, use the Search Bar fields.

   The search bar fields use pattern matching with operators such as "contains". For example, "john" will match john_smith, john42, or big_john.
9. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing VPN Usage Reports

VPN Usage reports provide information on the amount of VPN usage that occurs through the selected SonicWALL appliance(s).

VPN Usage reports can be used to view VPN usage by the hour, day, or over a period of days. Additionally, you can view the top users of VPN.

General bandwidth reports do not always provide a complete picture of network bandwidth usage. If a large amount of VPN traffic occurs, you might need to add bandwidth, upgrade network equipment, or reconfigure the VPN network.



**Note** All reports appear in the appliance's time zone.

Select from the following:

- To view a summary of the daily VPN bandwidth usage, see "Viewing the VPN Usage Summary Report" on page 239.
- To view the users who consume the most VPN bandwidth, see "Viewing the Top VPN Users" on page 241.
- To view VPN bandwidth usage over a period of time, see "Viewing VPN Usage Over Time" on page 242.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1483
Keysight, EX-1045
Page 1483

- To view the users who consume the most VPN bandwidth over time, see “Viewing VPN Usage Over Time” on page 242.
- To view the users who consume the most VPN bandwidth over time, see “Viewing the Top VPN Users Over Time” on page 243.
- To view VPN usage by policy, see “Viewing VPN Usage By Policy” on page 245.
- To view VPN usage by policy over time, see “Viewing the Top VPN Policies Over Time” on page 246.
- To view hourly VPN usage by policy, see “Viewing Hourly VPN Usage By Policy” on page 248.
- To view VPN services usage, see “Viewing the VPN Services Summary Report” on page 249.

# Viewing the VPN Usage Summary Report

The VPN Usage Summary report contains information on the number of VPN connections made through a SonicWALL appliance or all SonicWALL appliances during the specified day.

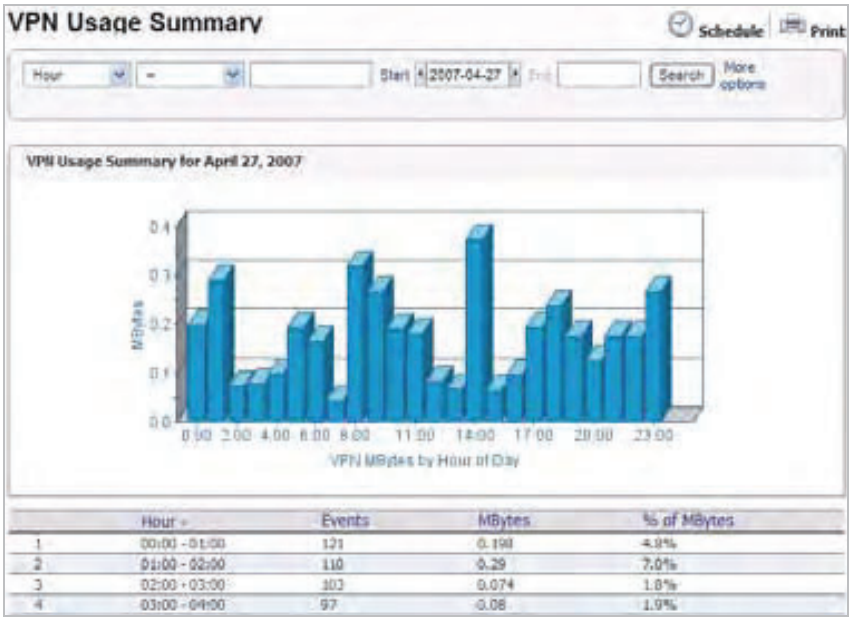
To view the VPN Usage Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1484
Keysight, EX-1045
Page 1484

3. Expand the **VPN Usage** tree and click **Summary**. The Summary page displays.




4. The bar graph displays the number of VPN connections made during each hour of the day.
5. The table contains the following information:
   - **Hour**—when the sample was taken.
   - **Events**—the number of mail events.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred by this user, compared to all users. For example, if 10,000 megabytes of data was transferred during the period and 2000 kilobytes was transferred by the top user, the **% of MBytes** field will display 20%.
6. The ViewPoint Reporting Module shows yesterday's report. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See "Managing Report Settings" on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



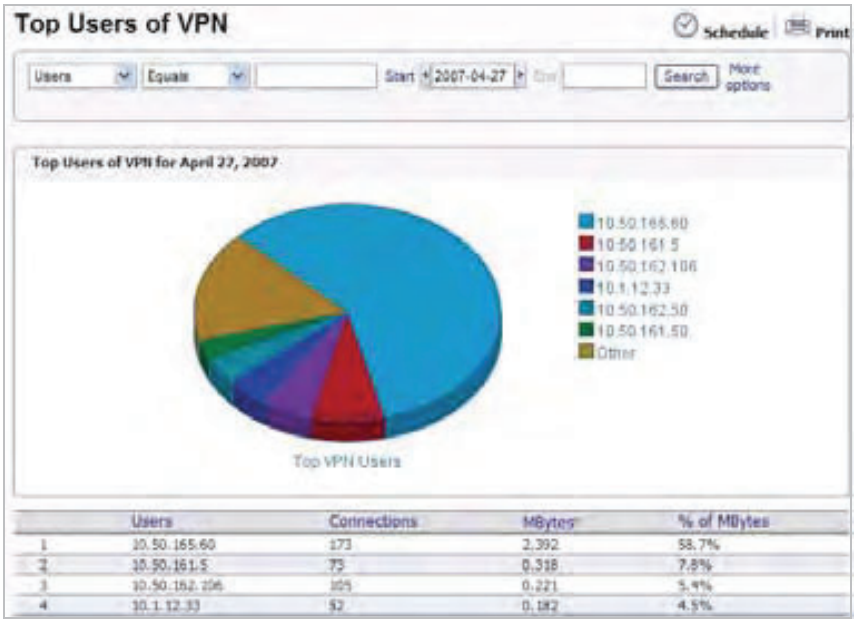
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1485
Keysight, EX-1045
Page 1485

# Viewing the Top VPN Users

The Top Users report displays the users who made the most VPN connections on the specified date.

To view the Top Users report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **VPN Usage** tree and click **Top Users**. The Top Users page displays.




4. The pie chart displays the VPN connections for the top VPN users.
5. The table contains the following information:
   - **Users**—the IP address of the user.
   - **Connections**—the number of VPN connections.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred by this user, compared to all users. For example, if 10,000 megabytes of data was transferred during the period and 2000 kilobytes was transferred by the top user, the **% of MBytes** field will display 20%.
6. By default, the ViewPoint Reporting Module shows yesterday's report, a pie chart, and the ten top users. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1486
Keysight, EX-1045
Page 1486

7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Rows per Screen**

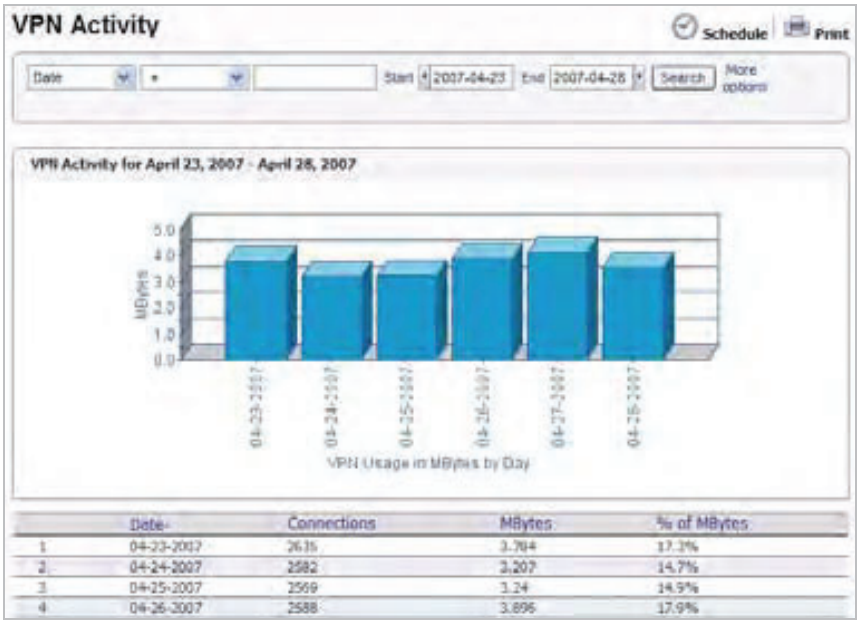
   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.
9. These settings will stay in effect for all similar reports during your active login session.

# Viewing VPN Usage Over Time

The VPN Usage Over Time report displays the daily number of VPN connections made through a SonicWALL appliance or all SonicWALL appliances during the specified time period.

To view the VPN Usage Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **VPN Usage** tree and click **Over Time**. The Over Time page displays.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1487
Keysight, EX-1045
Page 1487

4. The bar graph displays the number of VPN connections made during each day of the specified time period.
5. The table contains the following information:
   - **Date**—when the sample was taken.
   - **Connections**—the number of connections.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred by this user, compared to all users. For example, if 10,000 megabytes of data was transferred during the period and 2000 kilobytes was transferred by the top user, the **% of MBytes** field will display 20%.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing the Top VPN Users Over Time

The Top Users report displays the users who made the most VPN connections for the specified time period.

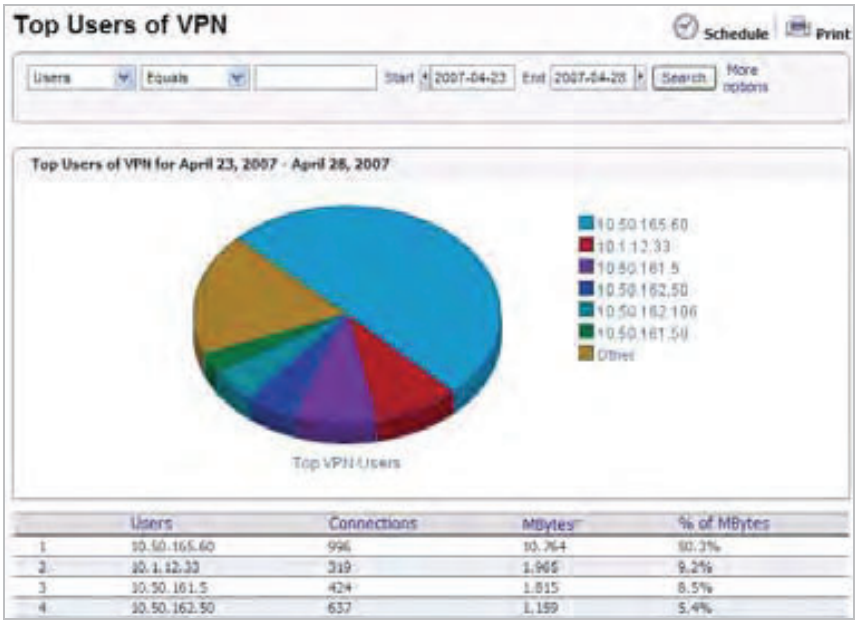
To view the Top Users report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1488
Keysight, EX-1045
Page 1488

3. Expand the **VPN Usage** tree and click **Top Users Over Time**. The Top Users Over Time page displays.




4. The pie chart displays the VPN connections for the top VPN users.
5. The table contains the following information:
   - **Users**—the IP address of the user.
   - **Connections**—the number of VPN connections.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred by this user, compared to all users. For example, if 10,000 megabytes of data was transferred during the period and 2000 kilobytes was transferred by the top user, the **% of MBytes** field will display 20%.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Users**
   - **Rows per Screen**

See “Managing Report Settings” on page 154.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1489
Keysight, EX-1045
Page 1489

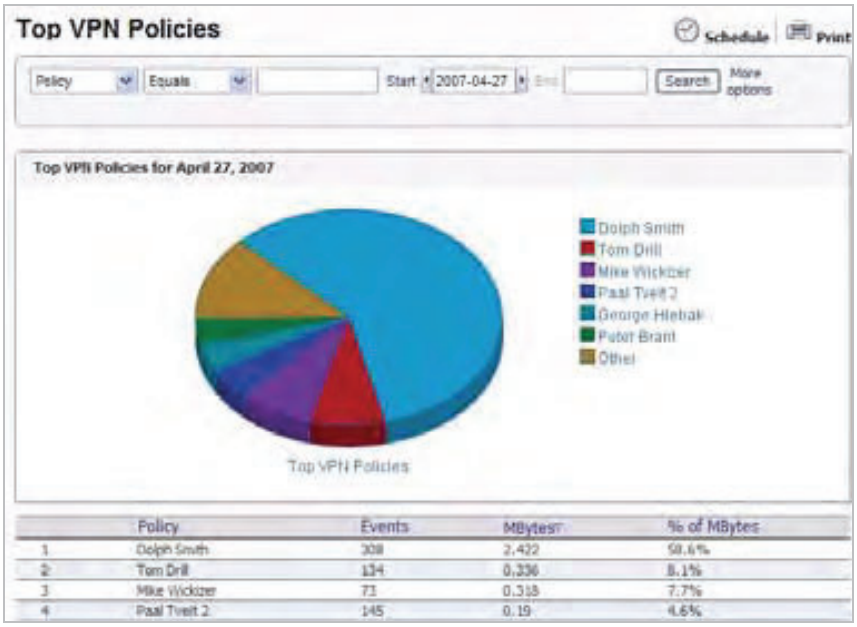
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing VPN Usage By Policy

The VPN Usage By Policy report contains information on VPN usage for a SonicWALL appliance, organized by policy.

To view the VPN Usage By Policy report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **VPN Usage** tree and click **By Policy**. The By Policy page displays.




4. The pie chart displays the amount of data transferred for each policy.
5. The table contains the following information:
   - **Policy**—the name of the policy.
   - **Events**—the number of VPN events.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred for this policy, compared to all other policies. For example, if a total of 10,000 megabytes was transferred and 2,500 megabytes was transferred for one policy, the **% of Usage** field will display 25%.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1490
Keysight, EX-1045
Page 1490

6. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing the Top VPN Policies Over Time

The By Policy Over Time report displays the top VPN Policies for the specified time period.

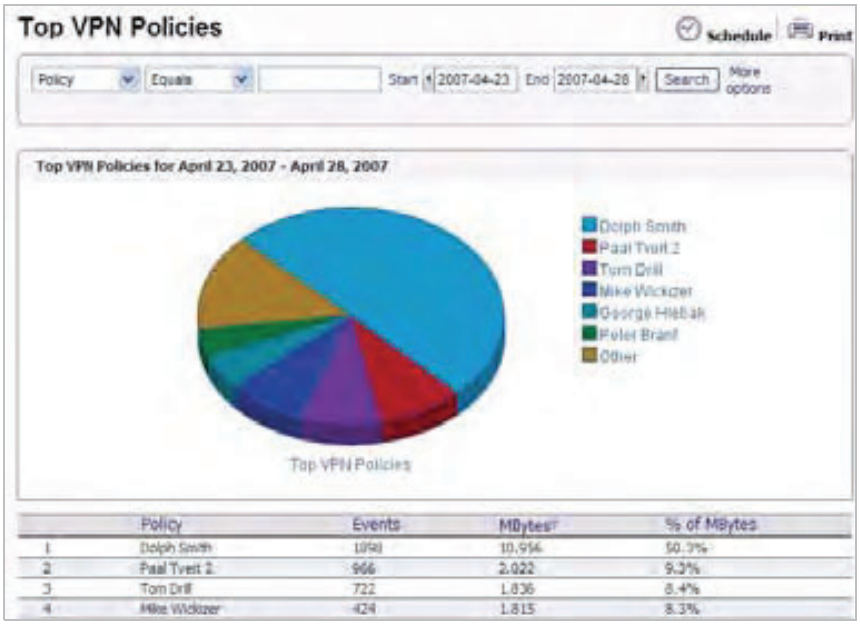
To view the By Policy Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1491
Keysight, EX-1045
Page 1491

3. Expand the **VPN Usage** tree and click **By Policy Over Time**. The By Policy Over Time page displays.




4. The pie chart displays the VPN connections for the top policies.
5. The table contains the following information:
   - **Policy**—the name of the policy.
   - **Events**—the number of VPN events.
   - **MBytes**—the number of megabytes transferred.
   - **% of MBytes**—the percentage of megabytes transferred for this policy, compared to all other policies for the period. For example, if a total of 100,000 megabytes was transferred and 3,000 megabytes was transferred for one policy, the **% of MBytes** field will display 3%.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1492
Keysight, EX-1045
Page 1492

7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Rows per Screen**

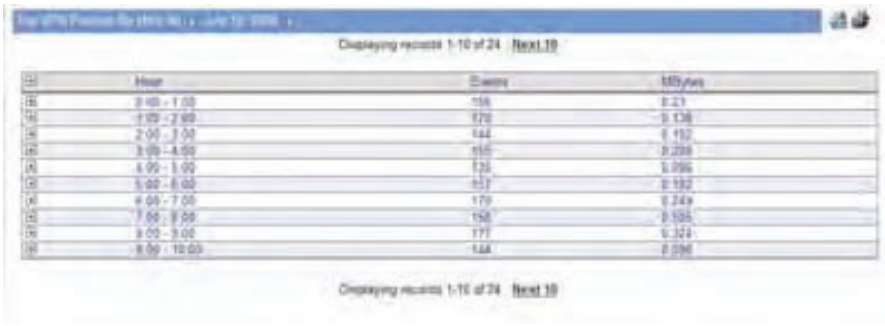
   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing Hourly VPN Usage By Policy

The VPN Usage By Policy Hourly report contains information on hourly VPN usage for a SonicWALL appliance, organized by policy.

To view the VPN Usage By Policy Hourly report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **VPN Usage** tree and click **By Policy Hourly**. The By Policy Hourly page displays.



4. The table contains the following information:
   - **Hour**—the period of time.
   - **Events**—the number of VPN events.
   - **MBytes**—the number of megabytes transferred.
5. The ViewPoint Reporting Module shows yesterday’s report. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1493
Keysight, EX-1045
Page 1493

6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart
   - **Hour Begin**
   - **Hour End**

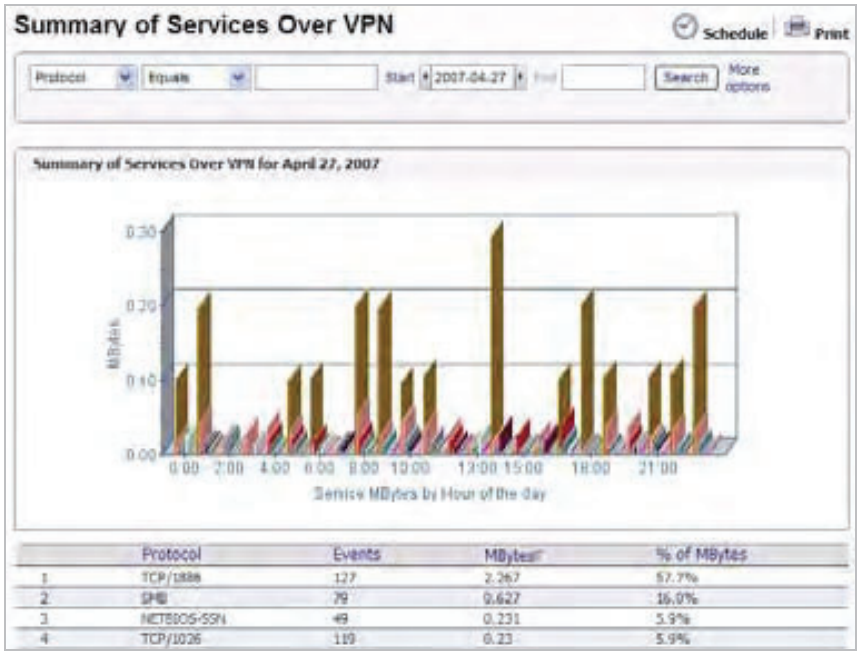
   See "Managing Report Settings" on page 154.
7. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing the VPN Services Summary Report

The Services Summary report displays the amount of traffic handled by each service during each hour of the specified day.

To view the Services Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **VPN Usage** tree and click **By Service**. The By Service page displays.




4. The bar graph displays the amount of bandwidth used by each service during each hour of the day.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1494
Keysight, EX-1045
Page 1494

5. The table contains the following information:
   - **Protocol**—the service.
   - **Events**—the number of events or “hits.”
   - **MBytes**—the number of megabytes.
   - **% of MBytes**—the percentage of megabytes transferred by this service on the selected day, compared to all other services. For example, if 1,000 megabytes were transferred and 900 megabytes were handled by the HTTP service, the **% of Mbytes** field will display 90%.
6. The ViewPoint Reporting Module shows yesterday’s report. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.
9. These settings will stay in effect for all similar reports during your active login session.

# Viewing Attacks Reports

Attacks reports show the number of attacks that were directed at or through the selected SonicWALL appliance(s). These include denial of service attacks, intrusions, probes, and all other malicious activity directed at the SonicWALL appliance or computers on the LAN or DMZ.



**Note** All reports appear in the appliance’s time zone.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1495
Keysight, EX-1045
Page 1495

Select from the following:

- To view a summary of the attacks, see “Viewing the Attack Summary Report” on page 251.
- To view the attacks by attack category, see “Viewing the Attacks By Category” on page 253.
- To view the attacks by source IP address, see “Viewing the Errors Report” on page 254.
- To view a summary of the errors and exceptions, see “Viewing the Errors Report” on page 254.
- To view attacks over a period of time, see “Viewing Attack Reports Over Time” on page 256.
- To view errors and exceptions over a period of time, see “Viewing Errors Over Time” on page 258.

# Viewing the Attack Summary Report

The Attack Summary report contains information on the number of attacks attempted on a SonicWALL appliance or all SonicWALL appliances during the specified day.

To view the Attack Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1496
Keysight, EX-1045
Page 1496

3. Expand the **Attacks** tree and click **Summary**. The Summary page displays.




4. The bar graph displays the number of attacks attempted during each hour of the day. The table contains the following information:
   - **Hour**—when the sample was taken.
   - **Attacks**—the number of attack attempts.
   - **% of Attacks**—the percentage of attacks during this hour, compared to the day. For example, if 1,000 attacks occurred during the day and 100 attacks occurred during the 2:00 time period, the **% of Attacks** field will display 10%.
5. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See "Managing Report Settings" on page 154.
7. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.



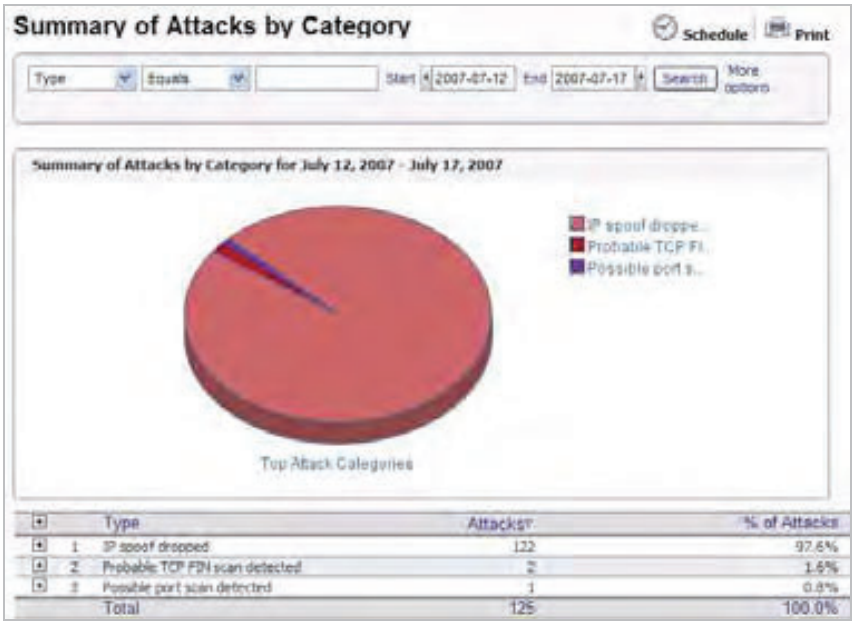
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1497
Keysight, EX-1045
Page 1497

# Viewing the Attacks By Category

The Attacks By Category report displays the attacks that occurred on the specified date, sorted by category.

To view the Attacks By Category report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Attacks** tree and click **By Category**. The By Category page displays.




4. The pie chart displays the percentage of each type of attack. To view source and destination information on the individual attacks, expand the category tree (indicated by a '+' sign).
5. The table contains the following information:
   - **Type**—the type of attack
   - **Source**—the IP address of the source
   - **Destination**—the IP address to the destination
     Click the highlighted source or destination IP address to access the Who is Source Website.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1498
Keysight, EX-1045
Page 1498

- **Attacks**—the number of attacks
- **% of Attacks**—the percentage of this type of attack, compared to all other attack types. For example, if 5,000 attacks occurred during the day and the IP Spoof makes up 500 of the attacks, its **% of Attacks** field will display 10%.

6. By default, the ViewPoint Reporting Module shows yesterday's report, a pie chart, and the ten top categories. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.
9. These settings will stay in effect for all similar reports during your active login session.

# Viewing the Errors Report

The Errors Summary report contains information on the number of dropped packets on a SonicWALL appliance or all SonicWALL appliances during the specified day.

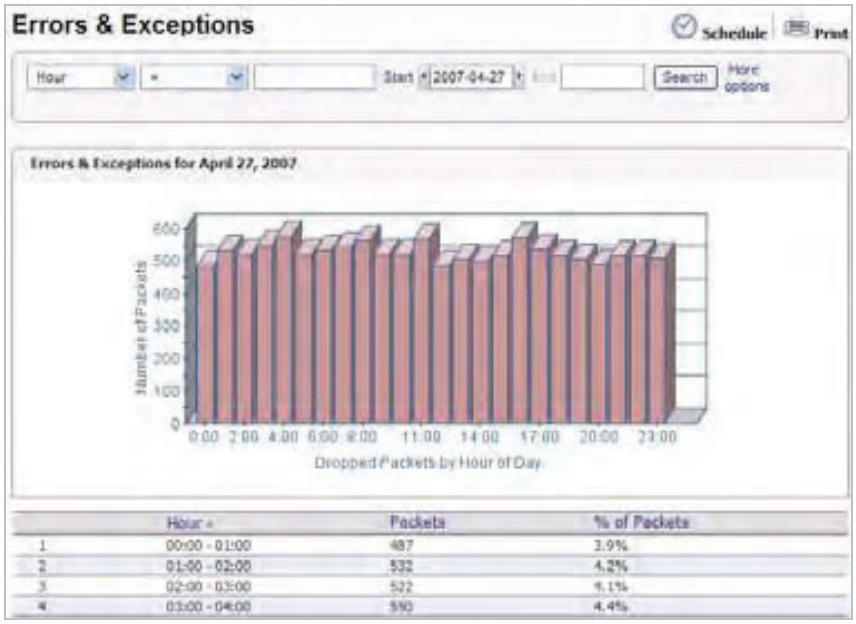
To view the Errors report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1499
Keysight, EX-1045
Page 1499

3. Expand the **Attacks** tree and click **Errors**. The Errors page displays.




4. The bar graph displays the packets that were dropped during each hour of the day.
5. The table contains the following information:
   - **Hour**—when the sample was taken.
   - **Packets**—the number of dropped packets.
   - **% of Packets**—the percentage of packets dropped during this hour, compared to the day. For example, if 1,000 packets were dropped during the day and 100 packets were dropped during the 1:00 time period, the **% of Packets** field will display 10%.
6. The ViewPoint Reporting Module shows yesterday's report.To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See "Managing Report Settings" on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.



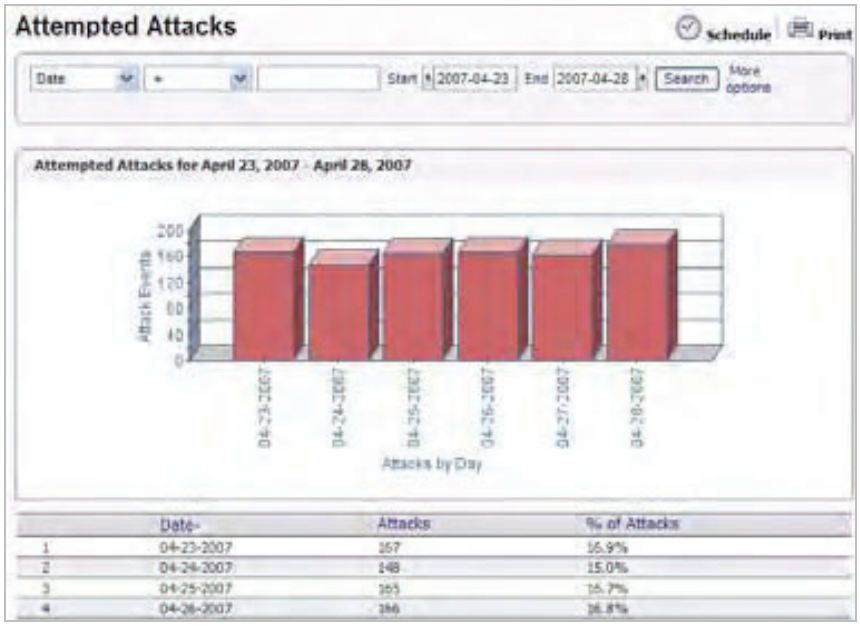
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1500
Keysight, EX-1045
Page 1500

# Viewing Attack Reports Over Time

The Attacks Over Time report displays the daily number of attempted attacks during the specified time period.

To view the Attacks Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Attacks** tree and click **Attacks Over Time**. The Attacks Over Time page displays.




4. The bar graph displays the number of attacks attempted each day of the time period.
5. The table contains the following information:
   - **Date**—when the sample was taken.
   - **Attacks**—the number of attacks.
   - **% of Attacks**—the percentage of attacks on this day, compared to the time period. For example, if 10,000 attacks occurred during the time period and 1,000 attacks occurred on Thursday, its **% of Attacks** field will display 10%.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1501
Keysight, EX-1045
Page 1501

7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing the Attacks By Category Over Time

The Categories Over Time report displays the number of attacks in each attack category during the specified time period.

To view the Categories Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Attacks** tree and click **Categories Over Time**. The Categories Over Time page displays.




4. The bar graph displays the number of attacks attempted each day of the specified time period. To view source and destination information on the individual attacks, expand the category tree (indicated by a ‘+’ sign).
5. The table contains the following information:
   - **Type**—the type of attack
   - **Source**—the IP address of the source



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1502
Keysight, EX-1045
Page 1502

- **Destination**—the IP address to the destination
  Click the highlighted source or destination IP address to access the Whois Source Website.
- **Attacks**—the number of attacks
- **% of Attacks**—the percentage of this type of attack, compared to all other attack types. For example, if 5,000 attacks occurred during the day and the IP Spoof makes up 500 of the attacks, its **% of Attacks** field will display 10%.

6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing Errors Over Time

The Errors Over Time report displays the number of errors during the specified time period.

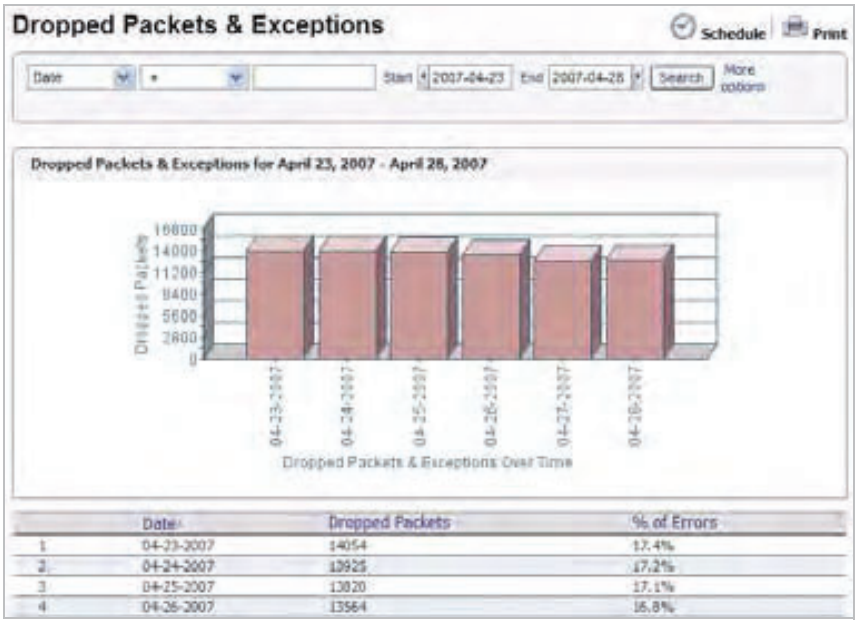
To view the Errors Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1503
Keysight, EX-1045
Page 1503

3. Expand the **Attacks** tree and click **Errors Over Time**. The Dropped Packets & Exceptions page displays.




4. The bar graph displays the number of packets that were dropped during each day of the specified time period.
5. The table contains the following information:
   - **Date**—when the sample was taken.
   - **Dropped Packets**—the number of dropped packets.
   - **% of Errors**—the percentage of dropped packets on this day, compared to the time period. For example, if 10,000 packets were dropped during the time period and 1,000 packets were dropped on Wednesday, its **% of Attacks** field will display 10%.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1504
Keysight, EX-1045
Page 1504

# Viewing Virus Attacks Reports

Virus Attacks reports show the number of virus attacks that were directed at or through the selected SonicWALL appliance(s).

**Note** All reports appear in the appliance's time zone.

If the selected appliance is not licensed for SonicWALL Gateway Anti-Virus, a sample report is displayed, as shown below. You can click the **Click Here** link near the top to view the global dashboard report showing all viruses and similar attacks currently being monitored by SonicWALL, or click the link at the bottom of the page to read detailed information about SonicWALL Gateway Anti-Virus and other subscription services.




Select from the following reports:

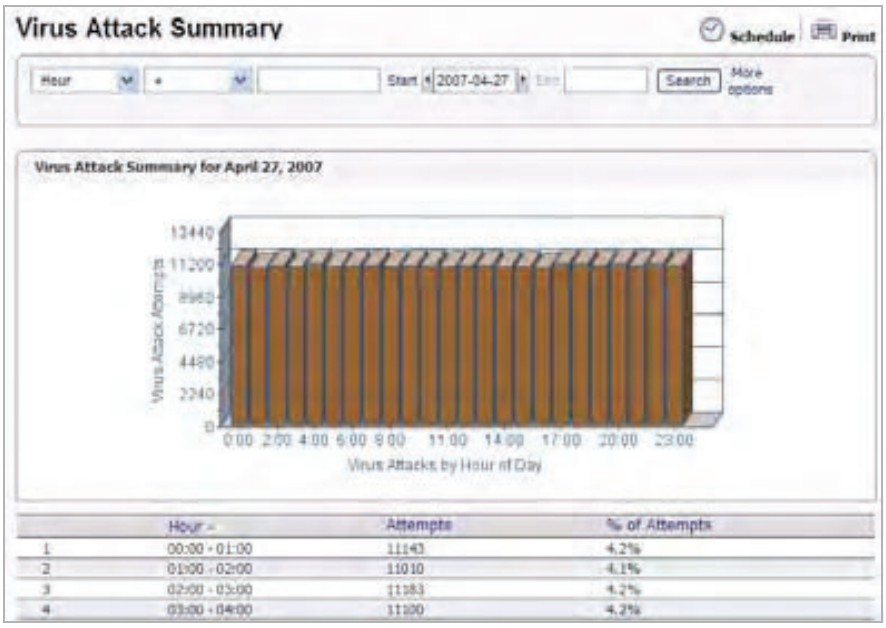
- To view the top virus, see "Viewing the Top Viruses By Attack Attempts Report" on page 262.
- To view the virus attacks by top destinations, see "Viewing the Virus Attack Attempts Report" on page 263.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1505
Keysight, EX-1045
Page 1505

- To view virus attacks over time, see “Viewing the Virus Attack Attempts Report” on page 263.
- To view virus attacks over a period of time, see “Viewing the Virus Attacks By User Report” on page 265.
- To view virus attacks by top destinations over time, see “Viewing Anti-Spyware Reports” on page 266.

9. Expand the **Virus Attacks** tree and click **Summary**. The Summary page displays




10. The bar graph displays the number of virus attacks attempted during each hour of the day. The table contains the following information:
    - **Hour**—the hour of the day for which the summary is provided.
    - **Attempts**—the number of times the virus attempted to infect the device during a pre-set time interval (the hour of the day is the default).
    - **% of Attempts**—the percent of attempts the current virus entry comprises as a portion of the aggregate number of virus attempts on the device during a pre-set time interval (the hour of the day is the default).
11. The ViewPoint Reporting Module shows yesterday’s report. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
12. Under Report Display Settings you can set:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1506
Keysight, EX-1045
Page 1506

- **Display Type**: **Chart and Table**, or **Table Only**
- **Chart Type**: **Area**, **Bar** or **Plot** chart

See “Managing Report Settings” on page 154.

13. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing the Top Viruses By Attack Attempts Report

The Top Viruses By Attack Attempts report displays the top viruses for the specified date.

To view the Top Viruses, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Virus Attacks** tree and click **By Virus**. The Top Viruses By Attack Attempts page displays.




4. The pie chart displays the percentage of virus attacks attempted in a given day.
5. The table contains the following information:
   - **Virus**—the name of the virus.
   - **Attempts**—the number of attack attempts.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1507
Keysight, EX-1045
Page 1507

- **% of Attempts**—the percentage of attempts as compared to the day.

6. The ViewPoint Reporting Module shows yesterday's report. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing the Virus Attack Attempts Report

The Virus Attack Attempts report displays the number of virus attempts over the specified time range.

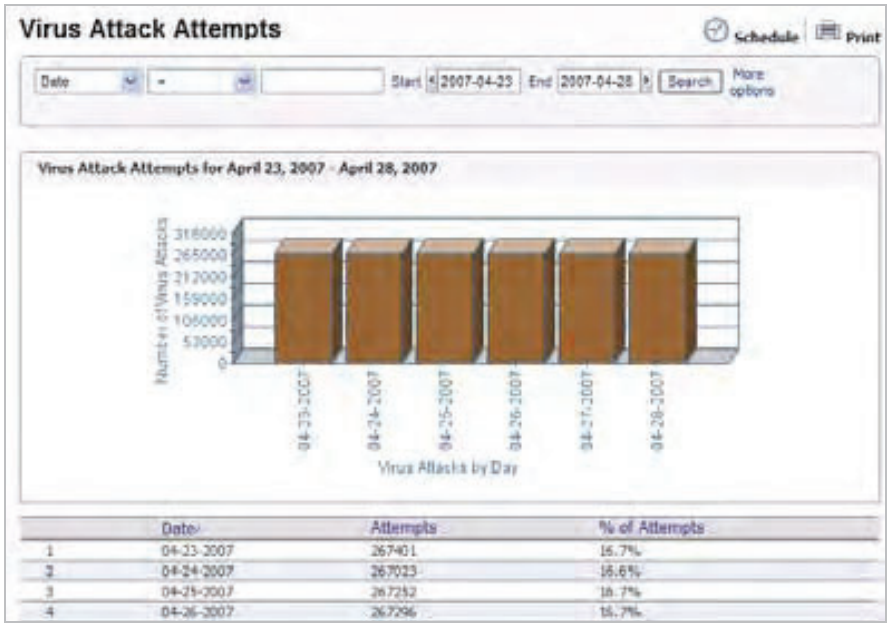
To view the Virus Attack Attempts report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1508
Keysight, EX-1045
Page 1508

3. Expand the **Virus Attacks** tree and click **Over Time**. The Virus Attack Attempts page displays.




4. The bar graph displays the number of virus attempts that were made during each day over a specified time period.
5. The table contains the following information:
   - **Date**—the date of when the sample was taken.
   - **Attempts**—the number of attempted virus attacks.
   - **% of Attempts**—the percentage of attempted virus attacks in a day compared to the time period. For example, if 5,000 attempts were made during the time period and 500 were made on one day, its **% of Attempts** field will display 10%.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1509
Keysight, EX-1045
Page 1509

7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

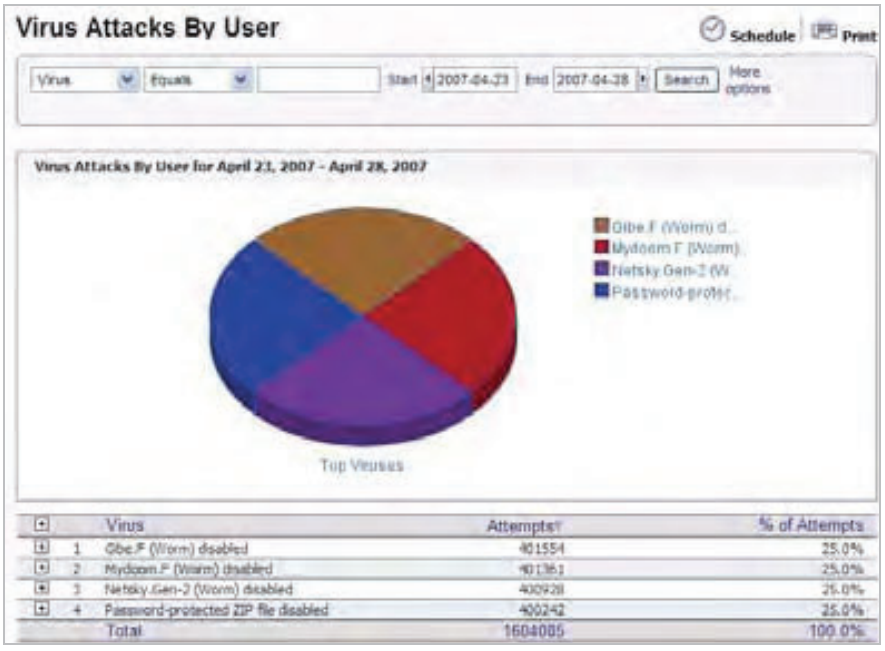
   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing the Virus Attacks By User Report

The Virus Attacks By User report displays the number of virus attack attempts over the specified time range.

To view the Virus Attacks By User report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Virus Attacks** tree and click **By Viruses Over Time**. The Virus Attacks By User page displays.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1510
Keysight, EX-1045
Page 1510

4. The pie chart displays the percentage of virus attacks attempted in a given day.
5. The table contains the following information:
   - **Virus**—the name of the virus.
   - **Attempts**—the number of attack attempts.
   - **% of Attempts**—the percentage of attempts compared to the day.
6. The ViewPoint Reporting Module shows yesterday's report. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing Anti-Spyware Reports

SonicWALL Anti-Spyware is included within the SonicWALL Gateway Anti-Virus (GAV), Anti-Spyware and Intrusion Prevention Service (IPS) unified threat management (UTM) solution. SonicWALL UTM delivers a comprehensive, real-time gateway security solution for your entire network.

Unlike other threat management solutions, SonicWALL Gateway Anti-Virus, Anti-Spyware and Intrusion Prevention Service has the capacity to analyze files of any size in real-time without the need to add expensive hardware drive or extra memory. SonicWALL Gateway Anti-Virus, Anti-Spyware and Intrusion Prevention Service includes a pro-active alerting mechanism that notifies network administrators when a new threat is discovered. Granular policy tools and an intuitive user interface enable administrators to configure a custom set of detection or prevention policies tailored to their specific network



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1511
Keysight, EX-1045
Page 1511

environment. Network administrators can create global policies between security zones and group attacks by priority, simplifying deployment and management across a distributed network.

If the selected appliance is not licensed for SonicWALL Anti-Spyware, a sample report is displayed, as shown below. You can click the **Click Here** link near the top to view the global dashboard report showing all spyware and similar attacks currently being monitored by SonicWALL, or click the link at the bottom of the page to read detailed information about SonicWALL Anti-Spyware and other subscription services.




See the following sections to view Anti-Spyware reports:

- “Viewing a Spyware Summary” on page 268
- “Viewing Spyware Attempts By Category” on page 269
- “Viewing Spyware Attempts Over Time” on page 270
- “Viewing Spyware Attempts By Category Over Time” on page 272



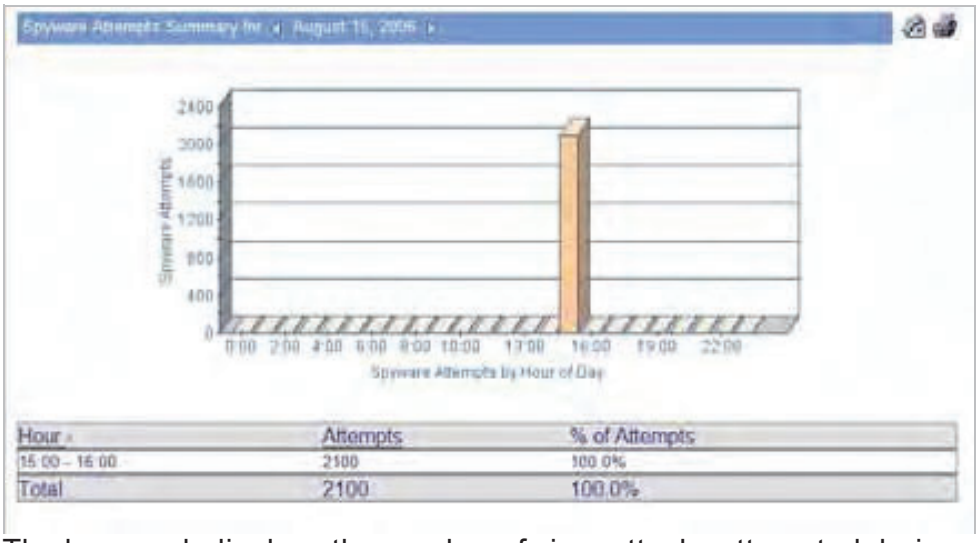
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1512
Keysight, EX-1045
Page 1512

# Viewing a Spyware Summary

The Anti-Spyware Summary report contains information on the number of spyware attempts by hour of the day.

To view a spyware Summary, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Anti-Spyware** tree and click **Summary**. The Summary page displays.




4. The bar graph displays the number of virus attacks attempted during each hour of the day.
5. The table contains the following information:
   - **Hour**—the hour of the day for which the summary is provided.
   - **Attempts**—the number of times the spyware attempted to infect the device during a pre-set time interval (the hour of the day is the default).
   - **% of Attempts**—the percent of attempts the current spyware entry comprises as a portion of the aggregate number of spyware attempts on the device during a pre-set time interval (the hour of the day is the default).
6. The ViewPoint Reporting Module shows yesterday's report. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1513
Keysight, EX-1045
Page 1513

7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

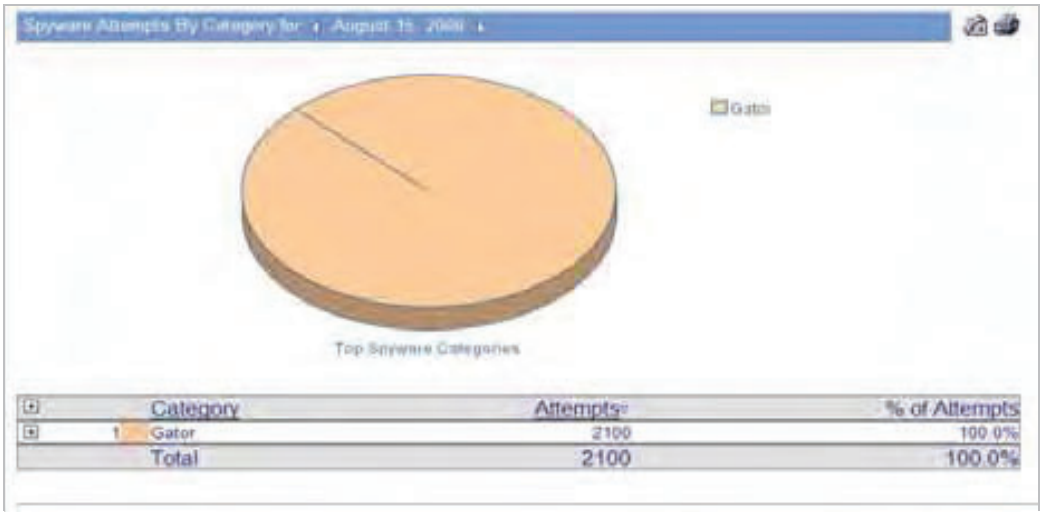
   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.
9. Note this page displays the number of spyware attempts that occurred during two-hour intervals during the past day.

# Viewing Spyware Attempts By Category

These reports display the spyware activity by category including the actual category or classification of the spyware, the priority, and the event/attacks type. By using the category as criteria, you can display details about the type/message text and number of events.

To view spyware attempts by category, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Anti-Spyware** tree and click **By Category**. The By Category page displays.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1514
Keysight, EX-1045
Page 1514

4. The pie chart displays the percentage of spyware attempts by category.
5. The table contains the following information:
   - **Category**—the category of the spyware.
   - **Attempts**—the number of times the spyware attempted to infect the device using the category as a criteria.
   - **% of Attempts**—the percent of attempts the current spyware entry comprises as a portion of the aggregate number of spyware attempts using the category as a criteria.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing Spyware Attempts Over Time

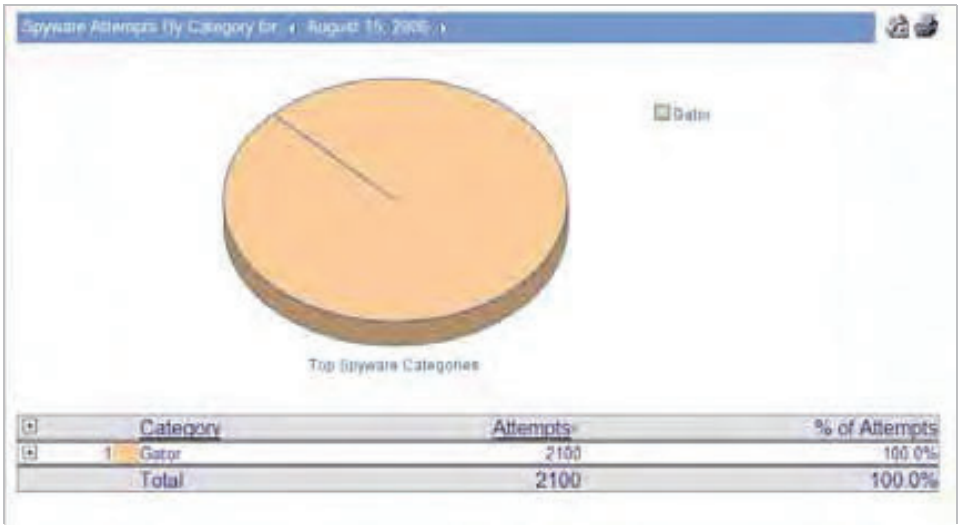
You can display spyware attempts over a set time interval. These reports are available at the unit and global levels similar to the other summary reports. To view spyware attempts using pre-set time intervals as the viewing criteria, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1515
Keysight, EX-1045
Page 1515

3. Expand the **Anti-Spyware** tree and click **Over Time**. The Over Time page displays.




4. The bar graph displays the number of spyware attempts that were made during each day over a specified time period.
5. The table contains the following information:
   - **Date**—the date for which the summary is provided.
   - **Attempts**—the number of times the spyware attempted to infect the device during a specific date.
   - **% of Attempts**—the percent of attempts the current spyware entry comprises as a portion of the aggregate number of spyware attempts on the device during a pre-set time interval.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.



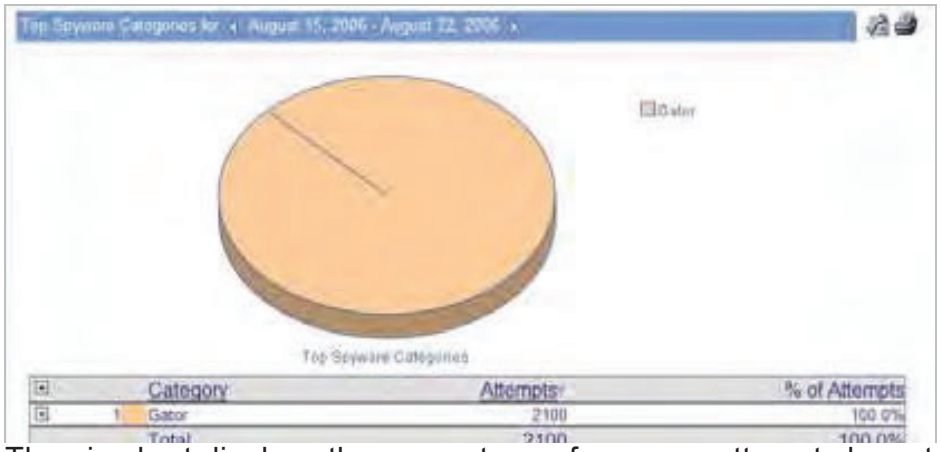
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1516
Keysight, EX-1045
Page 1516

# Viewing Spyware Attempts By Category Over Time

You can generate reports that display the spyware activity by category, such as the category, priority, and events/attacks over time. Using the category over time statistic as criteria for report generation provides details about the type/message text and number of events.

To view Anti-Spyware attempts using categories over time intervals as the viewing criteria, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Anti-Spyware** tree and click **By Category Over Time**. The By Category Over Time page displays.




4. The pie chart displays the percentage of spyware attempts by category. The table contains the following information:
   - **Category**—the category of the virus.
   - **Attempts**—the number of times the spyware attempted to infect the device during a pre-set time interval.
   - **% of Attempts**—the percent of attempts the current spyware entry comprises as a portion of the aggregate number of spyware attempts on the device during a pre-set time interval.
5. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1517
Keysight, EX-1045
Page 1517

6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
7. To display a limited group of items, use the Search Bar fields.



**Note** The search bar fields use pattern matching with operators such as “contains”. For example, “john” will match john_smith or john42.

8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing Intrusion Prevention Reports

The Intrusion Prevention Service (IPS) reports show the number of attempted intrusions that occurred during the specified time period.



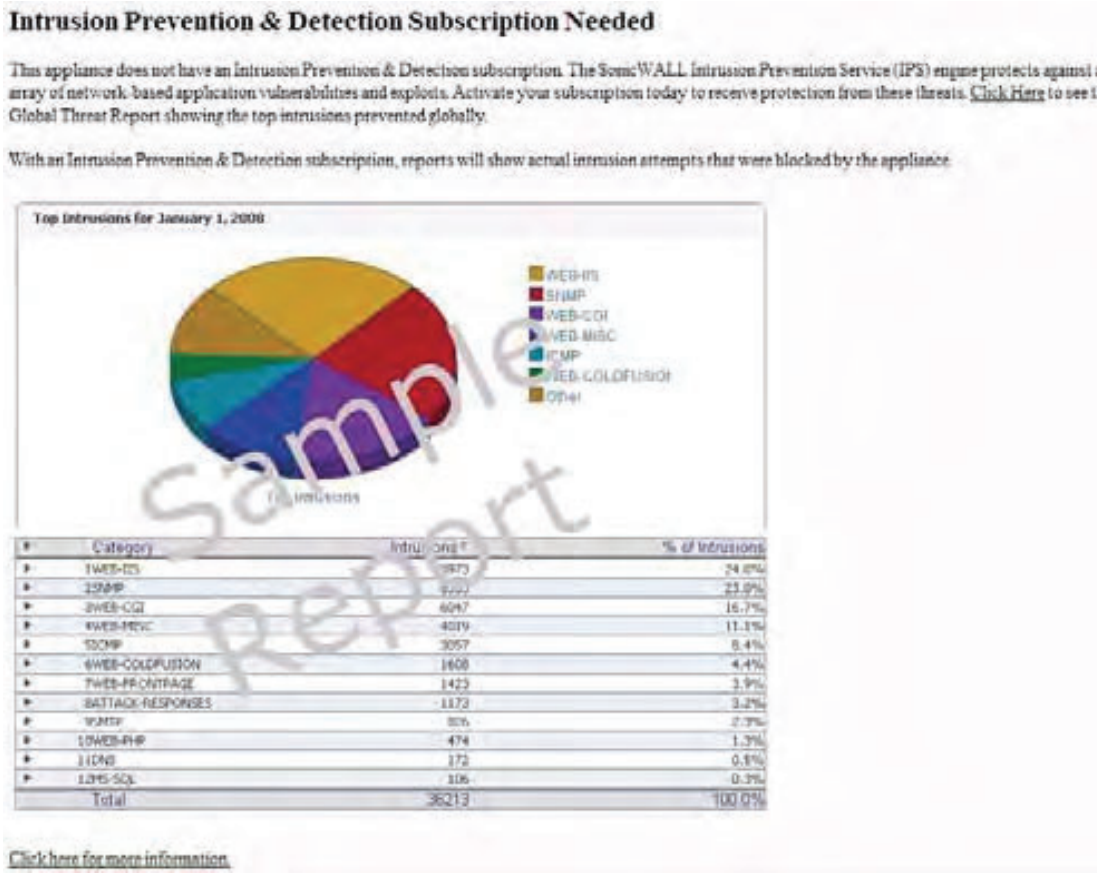
**Note** All reports appear in the appliance’s time zone.

If the selected appliance is not licensed for SonicWALL Intrusion Prevention Service, a sample report is displayed, as shown below. You can click the **Click Here** link near the top to view the global dashboard report showing all



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1518
Keysight, EX-1045
Page 1518

intrusions and similar attacks currently being monitored by SonicWALL, or click the link at the bottom of the page to read detailed information about SonicWALL Intrusion Prevention Service and other subscription services.




Select from the following intrusion reports:

- To view a summary of the attacks, see “Viewing the Intrusion Prevention Summary Report” on page 275.
- To view the attacks by source IP address, see “Viewing the Errors Report” on page 254.
- To view a summary of the errors and exceptions, see “Viewing the Errors Report” on page 254.
- To view attacks over a period of time, see “Viewing Attack Reports Over Time” on page 256.
- To view errors and exceptions over a period of time, see “Viewing Errors Over Time” on page 258.



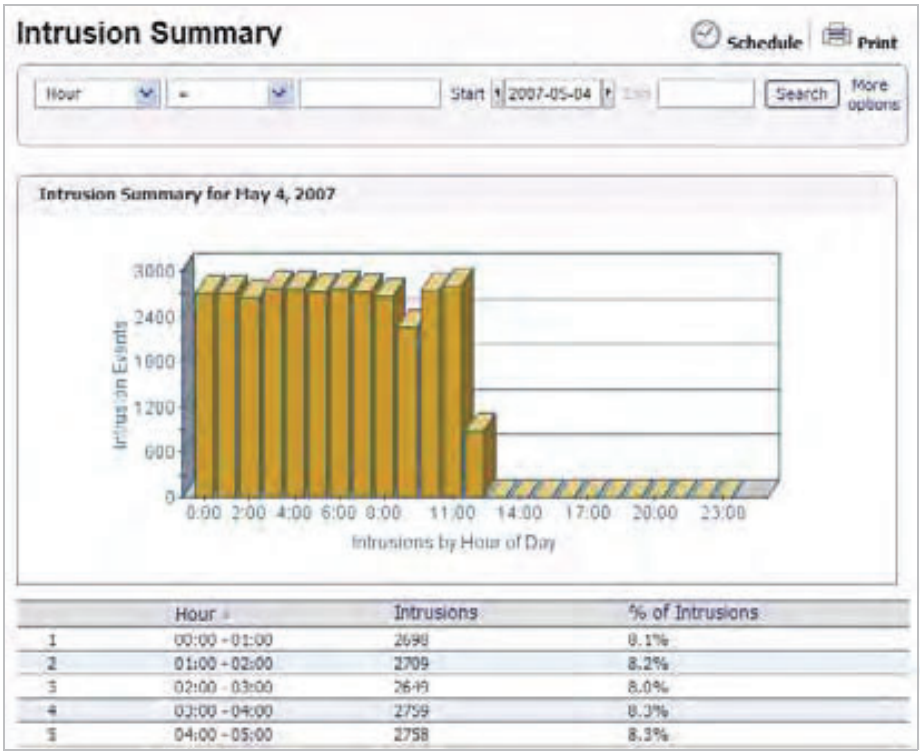
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1519
Keysight, EX-1045
Page 1519

# Viewing the Intrusion Prevention Summary Report

The Attack Summary report contains information on the number of attempted intrusions on a SonicWALL appliance or all SonicWALL appliances during the specified day.

To view the IPS Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Intrusion Prevention** tree and click **Summary**. The Summary page displays.




| | Hour | Intrusions | % of Intrusions |
|---|---|---|---|
| 1 | 00:00 - 01:00 | 2690 | 8.1% |
| 2 | 01:00 - 02:00 | 2709 | 8.2% |
| 3 | 02:00 - 03:00 | 2649 | 8.0% |
| 4 | 03:00 - 04:00 | 2759 | 8.3% |
| 5 | 04:00 - 05:00 | 2758 | 8.3% |

4. The bar graph displays the number of intrusions attempted during each hour of the day.
5. The table contains the following information:
   - **Hour**—when the sample was taken.
   - **Intrusions**—the number of intrusion attempts.
   - **% of Intrusions**—the percentage of intrusion attempts on this day, compared to the time period. For example, if 10,000 intrusion attempts occurred during the time period and 1,000 intrusion attempts occurred on Thursday, its **% of Intrusions** field will display 10%.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1520
Keysight, EX-1045
Page 1520

6. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See "Managing Report Settings" on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing Intrusion Attempts By Category

These reports display the intrusion activity by category including the actual category or classification of the intrusion, the priority, and the event/attacks type. By using the category as criteria, you can display details about the type/message text and number of events.

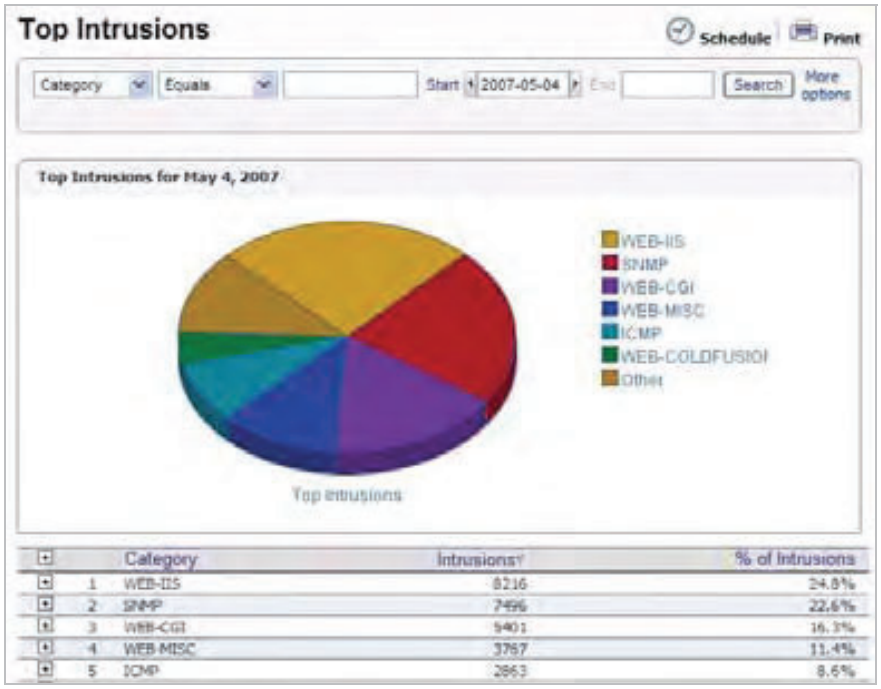
To view intrusion attempts by category, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1521
Keysight, EX-1045
Page 1521

3. Expand the **Intrusion Prevention** tree and click **By Category**. The By Category page displays.




4. The pie chart displays a list of intrusions attempted by category. The table contains the following information:
   - **Category**—the category of the intrusion attempt.
   - **Intrusions**—the number of intrusion attempts.
   - **% of Intrusions**—the percentage of intrusion attempts as a portion of the aggregate number of intrusion attempts using the category as a criteria.
5. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1522
Keysight, EX-1045
Page 1522

6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See “Managing Report Settings” on page 154.
7. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing Intrusions Over Time

The Over Time report displays the daily number of intrusion attempts during the specified time period.

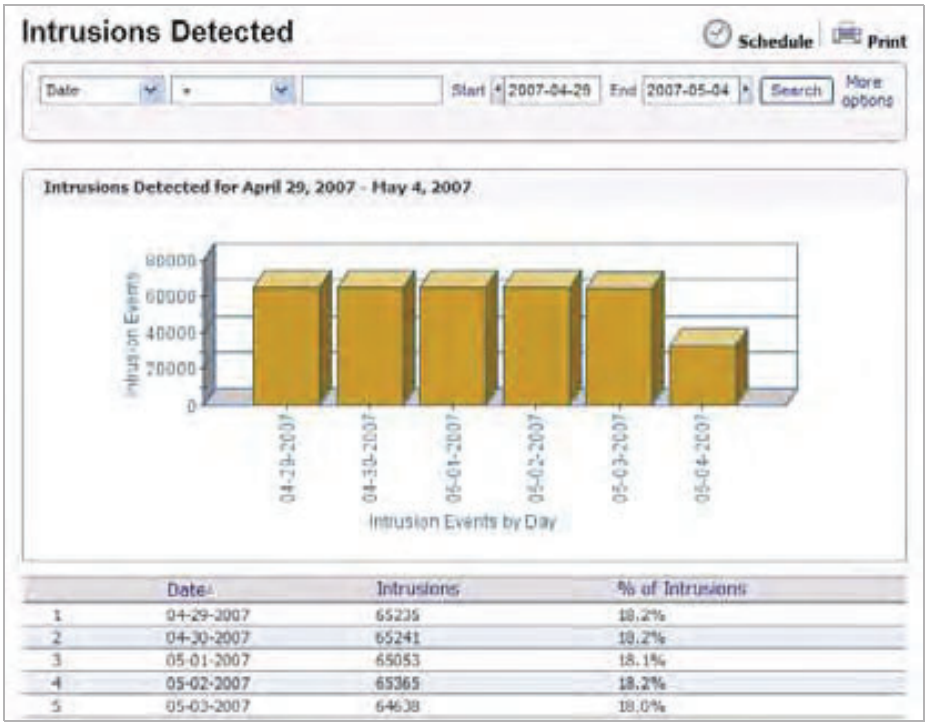
To view the Intrusions Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1523
Keysight, EX-1045
Page 1523

3. Expand the **Intrusion Prevention** tree and click **Intrusions Over Time**. The Intrusions Over Time page displays.




4. The bar graph displays the number of intrusions attempted each day of the specified time period.
5. The table contains the following information:
   - **Date**—when the sample was taken.
   - **Intrusions**—the number of intrusion attempts.
   - **% of Intrusions**—the percentage of intrusion attempts on this day, compared to the time period. For example, if 10,000 intrusion attempts occurred during the time period and 1,000 intrusion attempts occurred on Thursday, its **% of Intrusions** field will display 10%.
6. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
7. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar** or **Plot** chart

   See “Managing Report Settings” on page 154.
8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

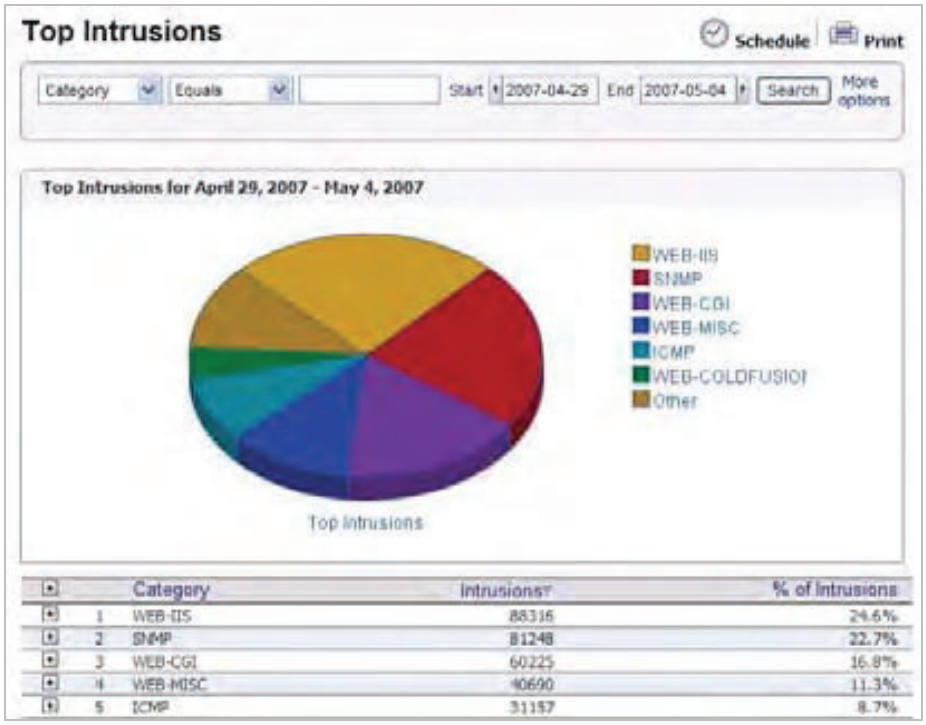


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1524
Keysight, EX-1045
Page 1524

# Viewing Intrusion Reports By Category Over Time

You can generate reports that display the intrusion activity by category, such as the category, priority, and events/attacks over time. Using the category over time statistic as criteria for report generation provides details about the type/message text and number of events. To view intrusion attempts using categories over time intervals as the viewing criteria, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Intrusion Prevention** tree and click **By Category Over Time**. The By Category Over Time page displays.




4. The pie chart displays a list of intrusions attempted by category over time. The table contains the following information:
   - **Category**—the category of the intrusion attempt.
   - **Intrusions**—the number of attempted intrusions during a pre-set time interval.
   - **% of Intrusions**—the percentage of intrusion attempts the current intrusion entry comprises as a portion of the aggregate number of intrusion attempts on the device during a pre-set time interval.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1525
Keysight, EX-1045
Page 1525

5. The ViewPoint Reporting Module shows yesterday's report. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar, or click **More Options** for report display settings.
6. Under Report Display Settings you can set:
   - **Display Type**: **Chart and Table**, or **Table Only**
   - **Chart Type**: **Area**, **Bar**, **Pie** or **Plot** chart
   - **Number of Items**
   - **Entries per Item**
   - **Rows per Screen**

   See "Managing Report Settings" on page 154.
7. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing Application Firewall Reports

Application Firewall reports provide information on the applications, users, and Application Firewall policies that are handled by Application Firewall on the selected SonicWALL appliance(s). The Application Firewall feature is available on SonicWALL NSA and SonicWALL TZ 210 Series appliances in SonicOS Enhanced 5.0 and higher.

Application Firewall reports can be used to view Application Firewall usage by the day or over a period of days. Additionally, you can view the top applications, top users, or top policies for Application Firewall on a single SonicWALL NSA or SonicWALL TZ 210 series appliance.

Clickable reports (Graphs and Data) are supported, providing drill-down reporting information by clicking the graphical elements (such as pie chart slices) and data rows. For example, you can drill down to the User report level by clicking a user in one of the Top reports.



**Note** All reports appear in the appliance's time zone.

Select from the following:

- To view a summary of the daily Application Firewall usage, see "Viewing the Application Firewall Summary Report" on page 282.
- To view Application Firewall usage over time, see "Viewing the Application Firewall Over Time Report" on page 283.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1526
Keysight, EX-1045
Page 1526

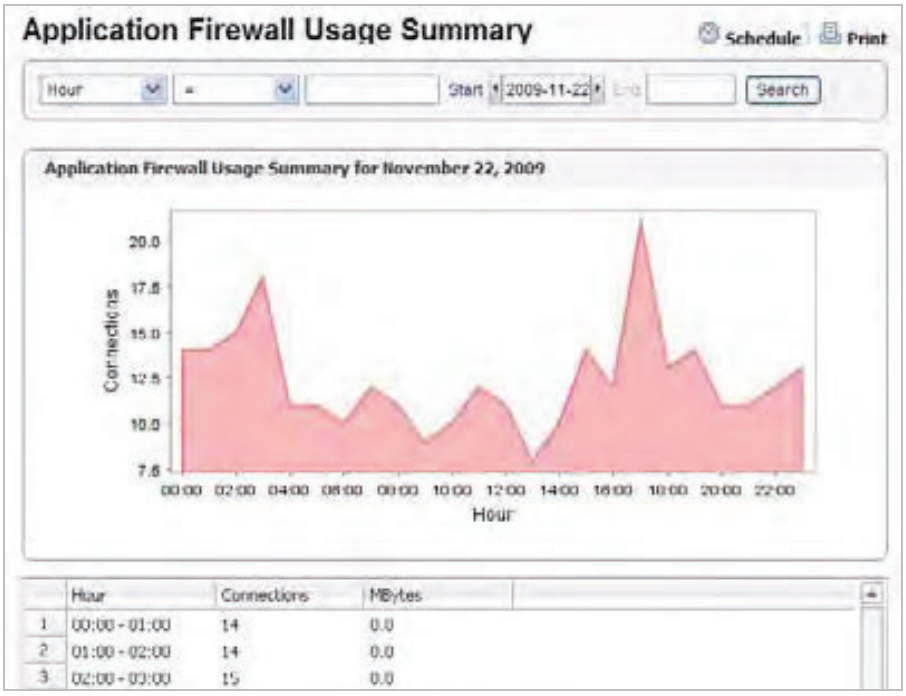
- To view the applications most often intercepted by Application Firewall, see “Viewing Application Firewall Top Applications” on page 284.
- To view the users whose traffic is most often intercepted by Application Firewall, see “Viewing Application Firewall Top Users” on page 285.
- To view the Application Firewall policies that are used the most, see “Viewing Application Firewall Top Policies” on page 286.

# Viewing the Application Firewall Summary Report

The Application Firewall Summary report contains information on the number of connections incurring Application Firewall activity logged by a SonicWALL appliance during each hour of the specified day, or at the global level, for all SonicWALL appliances for the day.

To view the Application Firewall Summary report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Application Firewall** tree and click **Summary**. The Summary page displays.




4. The table contains the following information:
   - **Hour**—when the sample was taken



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1527
Keysight, EX-1045
Page 1527

- **Connections**—number of attempted connections logged (and possibly blocked) by Application Firewall
- **Mbytes**—megabytes of data transferred during the connections

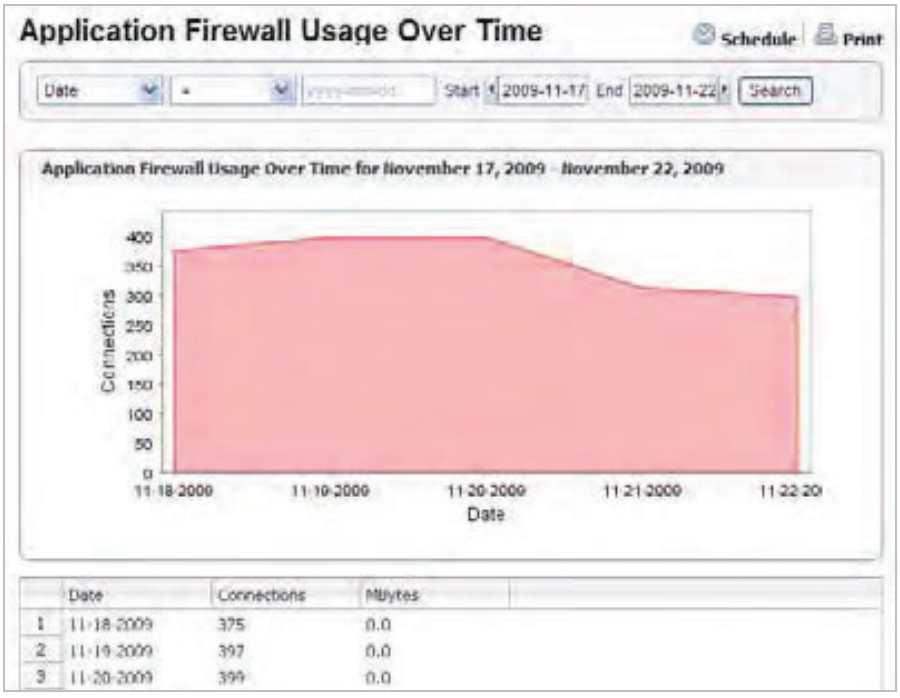
5. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report, click the **Start** and **End** fields to access the drop-down calendars, select the desired dates, and then click **Search**. The ViewPoint Reporting Module displays the report for the selected day or date range.

# Viewing the Application Firewall Over Time Report

The Application Firewall Over Time report displays the amount of Application Firewall usage handled by a SonicWALL appliance or a group of SonicWALL appliances for the specified time period.

To view the Application Firewall Over Time report, perform the following steps:

1. Click the **UTM** tab.
2. Select the global icon or a SonicWALL appliance.
3. Expand the **Application Firewall** tree and click **Over Time**. The Over Time page displays.




4. The table contains the following information:
   - **Date**—when the sample was taken



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1528
Keysight, EX-1045
Page 1528

- **Connections**—number of attempted connections logged (and possibly blocked) by Application Firewall
- **Mbytes**—megabytes of data transferred during the connections

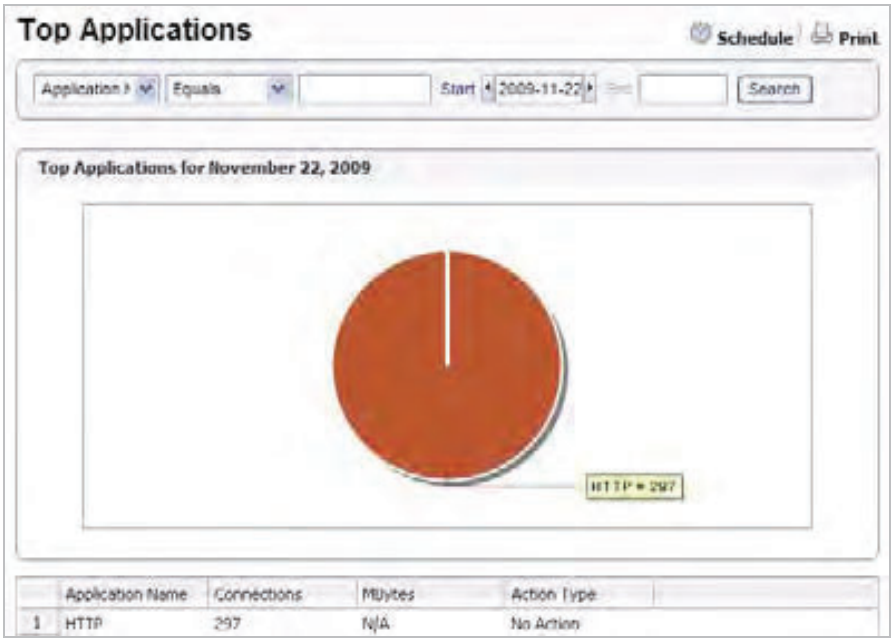
5. To change the date of the report, click the **Start** and **End** fields to access the drop-down calendars, select the desired dates, and then click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing Application Firewall Top Applications

The Top Applications report displays the applications that were most logged and/or blocked by Application Firewall on the specified date. The Top Applications report is available at the unit level.

To view the Top Applications report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Application Firewall** tree and click **Top Applications**. The Top Applications page displays.




4. The table contains the following information:
   - **Application Name**—the type of application, such as HTTP, FTP, and so on



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1529
Keysight, EX-1045
Page 1529

- **Connections**—number of attempted connections logged (and possibly blocked) by Application Firewall
- **Mbytes**—megabytes of data transferred during the connections
- **Action Type**—either No Action, Logged, or Blocked

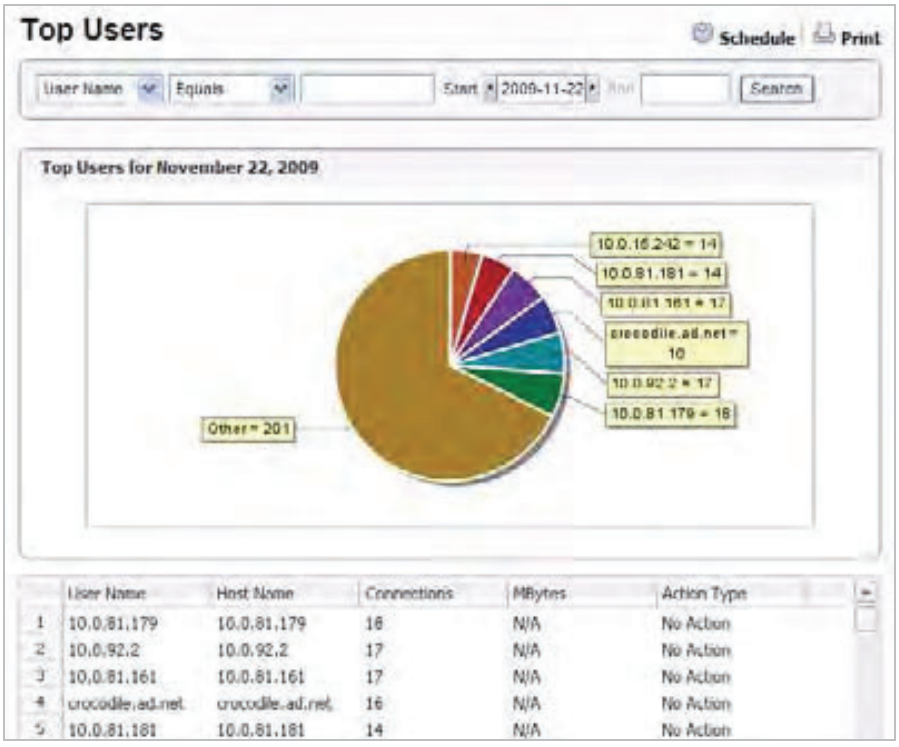
5. To change the date of the report, click the **Start** field to access the drop-down calendar, select the desired date, and then click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing Application Firewall Top Users

The Top Users report displays the users who made the most logged and/or blocked connections by Application Firewall on the specified date. The Top Users report is available at the unit level.

To view the Top Users report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Application Firewall** tree and click **Top Users**. The Top Users page displays.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1530
Keysight, EX-1045
Page 1530

4. The table contains the following information:
   - **User Name**—the user's name or IP address
   - **Host Name**—the host name or IP address of the computer that made the connection
   - **Connections**—number of attempted connections logged (and possibly blocked) by Application Firewall
   - **Mbytes**—megabytes of data transferred during the connections
   - **Action Type**—either No Action, Logged, or Blocked
5. To change the date of the report, click the **Start** field to access the drop-down calendar, select the desired date, and then click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing Application Firewall Top Policies

The Top Policies report displays the Application Firewall policies that were triggered the most on the specified date. The Top Policies report is available at the unit level.

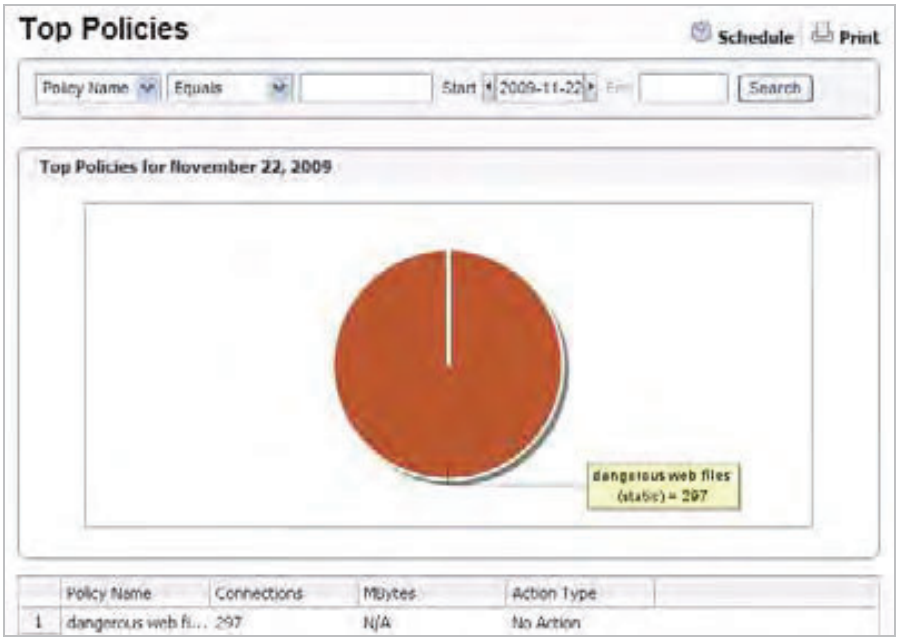
To view the Top Policies report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1531
Keysight, EX-1045
Page 1531

3. Expand the **Application Firewall** tree and click **Top Policies**. The Top Policies page displays.




4. The table contains the following information:
   - **Policy Name**—the Application Firewall policy name
   - **Connections**—number of attempted connections logged (and possibly blocked) by Application Firewall
   - **Mbytes**—megabytes of data transferred during the connections
   - **Action Type**—either No Action, Logged, or Blocked
5. To change the date of the report, click the **Start** field to access the drop-down calendar, select the desired date, and then click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing Authentication Reports

The login reports show user logins, administrator logins, and failed login attempts for users and administrators. Authentication reports are available at the unit level.

**Note** All reports appear in the appliance's time zone.

Select from the following:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1532
Keysight, EX-1045
Page 1532

- "Viewing the User Login Report" on page 288
- "Viewing the Administrator Login Report" on page 289
- "Viewing the Failed Login Report" on page 289


# Viewing the User Login Report

The user login report shows users that logged on to the SonicWALL appliance during the specified day to bypass content filtering or to remotely access local network resources.

To view the User Login report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Authentication** tree and click **User Login**. The User Login page displays.




4. The table contains the following information:
   - **User**—the user name.
   - **Time**—time the user logged in.
5. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar.

   See "Managing Report Settings" on page 154.
6. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1533
Keysight, EX-1045
Page 1533

# Viewing the Administrator Login Report

The administrator login report shows successful administrator logins during the specified day. This report is useful for identifying misuse and unauthorized management of a SonicWALL appliance.

To view the Admin Login report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Authentication** tree and click **Admin Login**. The Admin Login page displays.



4. The table contains the following information:
   - **User**—the user name.
   - **Time**—time the user logged in.
5. To change the date range of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar.

   See “Managing Report Settings” on page 154.
6. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date range.

# Viewing the Failed Login Report

The failed login report shows failed login attempts for users and administrators that attempted to log on to the SonicWALL appliance during the specified day. This report is useful for identifying unauthorized access attempts and potentially malicious activity.

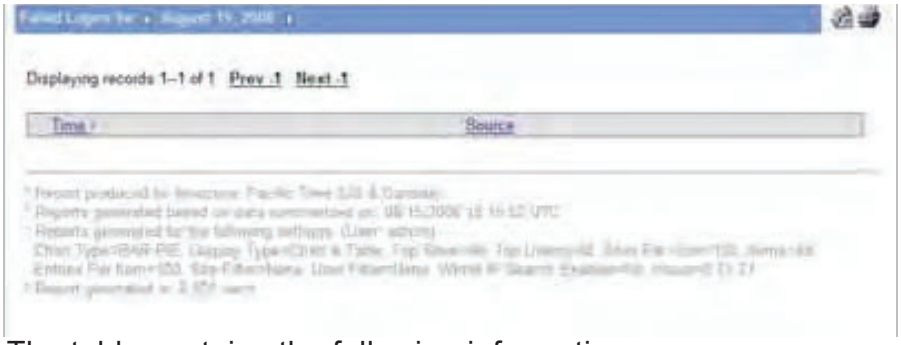
To view the Failed Login report, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1534
Keysight, EX-1045
Page 1534

3. Expand the **Authentication** tree and click **Failed Login**. The page displays.




4. The table contains the following information:
   - **User**—the user name.
   - **Time**—time the user logged in.
   - **IP Address**—IP address of the user.
5. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report, use the Search Bar and click the **Start** or **End** field to access the drop-down calendar.

   See "Managing Report Settings" on page 154.
6. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing the Log

The Log Viewer contains detailed information on each transaction that occurred on the SonicWALL appliance. This information is stored for the time that you specified in the configuration settings.

It is necessary to enable the Log Viewer for UTM or SSL-VPN appliances for which you wish to generate Custom Reports. See "Using Custom Reports on UTM Appliances" on page 163.

**Note** The Log Viewer displays raw log information for every connection. Depending on the amount of traffic, this can quickly consume a large amount of space in the database. It is highly recommended to be careful when choosing the number of days of information that will be stored. For more information, see "Scheduling and Configuring Reports" on page 133.

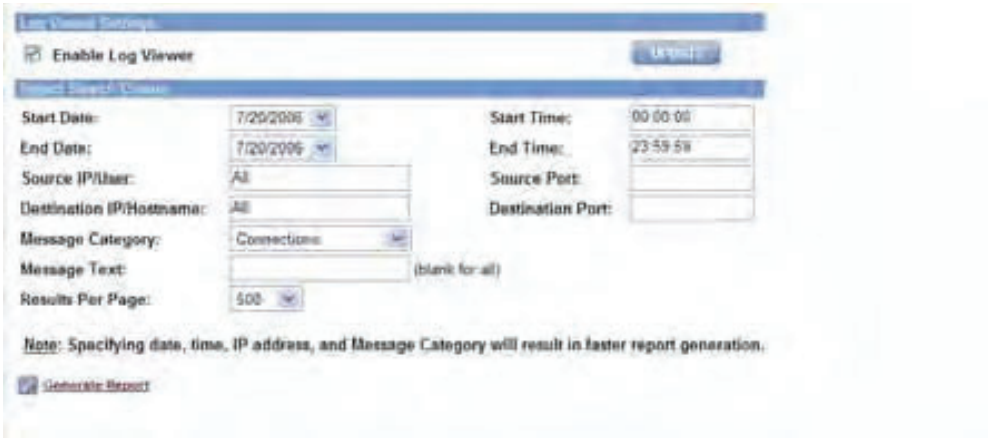


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1535
Keysight, EX-1045
Page 1535

# Viewing the Log for a SonicWALL Appliance

To view the Log, perform the following steps:

1. Click the **UTM** tab.
2. Select a SonicWALL appliance.
3. Expand the **Log Viewer** tree and click **Search**. The Search page displays.




4. Select **Enable Log Viewer** and then click **Update** to turn on collection of raw data in the database and enable viewing of that log data. This can consume a large amount of space in your database. Review your database space constraints before enabling the log viewer. The maximum number of appliances for which Log Viewer can be enabled is controlled on the Console > Reports > Settings page. See "Controlling the Number of Appliances with Log Viewer Enabled" on page 72.

**Note** Custom Reports are available on appliances with Log Viewer enabled. See "Using Custom Reports on UTM Appliances" on page 163.

5. Under Select Search Criteria, select the date range to view data from in the **Start Date** and **End Date** fields.
6. Enter the starting time of events to view in the **Start Time** field.
7. Enter the ending time of events to view in the **End Time** field.
8. To limit the report to data originating from specific IP addresses or users, enter the source IP address or user name in the **Source IP/User** field. To view all IP addresses, enter **All**.
9. To view log entries for data originating from a particular port, enter the port number in the **Source Port** field.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1536
Keysight, EX-1045
Page 1536

10. To limit the report to data going to specific IP addresses or hosts, enter the destination IP address or host name in the **Destination IP/Hostname** field. To view log entries for data going to all IP addresses, enter **All**.
11. To view log entries for data going to a particular port, enter the port number in the **Destination Port** field.
12. Select the type of events to view from the **Message Category** list box.
13. To limit the report to messages containing a specific text string, enter the text in the **Message Text** field. Leave the field blank to view all messages.
14. Select the number of entries to display per page from the **Results Per Page** field.
15. Click **Generate Report**. The Log Viewer Results page displays.




16. Search through the entries to find the information for which you are searching. To view the next page of entries, click **Next**.
17. To generate another report, click **Search** again in the **Log Viewer** tree.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1537
Keysight, EX-1045
Page 1537

# CHAPTER 16
# SSL-VPN Reporting

This chapter describes how to manage SonicWALL ViewPoint SSL-VPN reporting by customizing and defining scheduled reports and summarization for SSL-VPN appliances.

For details about viewing specific SSL-VPN reports, see “Viewing SSL-VPN Reports” on page 299.

This chapter contains the following sections:


- “SSL-VPN Reporting Overview” section on page 293
- “Using and Configuring SSL-VPN Reporting” section on page 295


## SSL-VPN Reporting Overview

This section provides an introduction to the SSL-VPN reporting feature. This section contains the following subsections:


- “What is SSL-VPN Reporting?” section on page 294
- “Benefits of SSL-VPN Reporting” section on page 294
- “How Does SSL-VPN Reporting Work?” section on page 295


After reading the ViewPoint SSL-VPN Reporting Overview section, you will understand the main steps to be taken in order to create and customize reports successfully.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1538
Keysight, EX-1045
Page 1538

# What is SSL-VPN Reporting?

SSL-VPN reporting allows you to configure and design the way you view your reports and the manner in which you receive them. This feature offers various types of static and dynamic reporting in which you can customize the way information is reported.

SonicWALL ViewPoint SSL-VPN reporting provides a visual presentation of all your configured report settings and information. With SSL-VPN reporting, you are able to view your reports in new enhanced graphs, create granular, custom reports, create scheduled reports, and search for reports using the search bar tool.

Custom reports are also available in SSL-VPN reporting. SonicWALL SSL-VPN appliances provide a Resource Activity custom report for tracking the source, destination, and other information about resource activity passing through a SonicWALL SSL-VPN device.

The Custom Reports feature provides an intuitive, responsive interface for customizing the report layout and configuring content filtering prior to generating the report. Two types of reports are available: Detailed Reports and Summary Reports. Both provide detailed information, but are formatted to meet different needs. A Detailed Report displays the data in sortable, resizable columns, while a Summary Report provides top level information in graphs that you can click to drill down for detailed information.

Once you set up a Custom Report that meets your needs, you can save your settings as a template for reuse, set a schedule to run the report, export the report as a PDF or CSV (Excel) file, or print report pages.

# Benefits of SSL-VPN Reporting

SSL-VPN reports provide visibility into the resource use by logged in users, leading to policies that enhance the user experience and the productivity of employees. The following capabilities contribute to the benefits of the SSL-VPN reporting feature:

- Custom reports can track events to the minute or second of the day for forensics and troubleshooting
- Interactive charts allow drill-down into specific details
- Table structure with ability to adjust column width of data grid
- Improved report navigation
- Report search
- Scheduled reports



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1539
Keysight, EX-1045
Page 1539

## How Does SSL-VPN Reporting Work?

SSL-VPN appliances send syslog data to the ViewPoint syslog collector, similar to SonicWALL UTM appliances. Once summarization takes place, you can create, schedule, view, and search for SSL-VPN reports from the ViewPoint central reporting interface.

SSL-VPN Custom Reports are based on raw syslog information contained in a database that is created daily from the raw syslog data sent from all managed or monitored appliances. This database is saved using a date/time suffix, and contains tables full of data for each appliance. All the syslog data received by SonicWALL ViewPoint is available in the database.

**Note** The raw syslog database required by Custom Reports is not enabled by default, as it is highly resource intensive. This functionality must be enabled per unit in the UTM > Log Viewer screen.

SSL-VPN Reporting supports scheduled reports to be sent on a daily, weekly, or monthly basis to any specified email address.

# Using and Configuring SSL-VPN Reporting

This section describes how to use and configure SSL-VPN reporting. See the following subsections:


- "About Viewing Available SSL-VPN Report Types" section on page 295
- "Configuring SSL-VPN Scheduled Reports" section on page 296


## About Viewing Available SSL-VPN Report Types

To view the available types of reports for SSL-VPN, perform the following steps:

1. Log into your ViewPoint management console.
2. Click the **SSL-VPN** tab.

   The SSL-VPN screen displays the following list of reports:

   **Node Level reports:**

   – **General**

   –Status: information about the appliance



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1540
Keysight, EX-1045
Page 1540

- **Bandwidth**
  - Summary: total connections listed by hour
  - Top Users: connections listed by user
  - Over Time: connections listed by date
  - Top Users Over Time: connections listed by user for the selected date range
- **Custom Report**
  - Resource Activity: source, destination, and other information about resource activity
- **Resources**
  - Summary: connections per connection protocol (HTTPS, NetExtender, etc)
  - Top Users: connections listed by user
- **Authentication**
  - User Login: user, time, and source of successful authentication-daily. User Login reports now combine admin users with all other users in the same report.
  - Failed login: time and source host of failed logins for one day

**Global Level Reports**:

- **General**
  - Status: number of units in the system and their ViewPoint license status
- **Bandwidth**
  - Summary: connections per SSL-VPN appliance
  - Over Time: total connections by date for group

# Configuring SSL-VPN Scheduled Reports

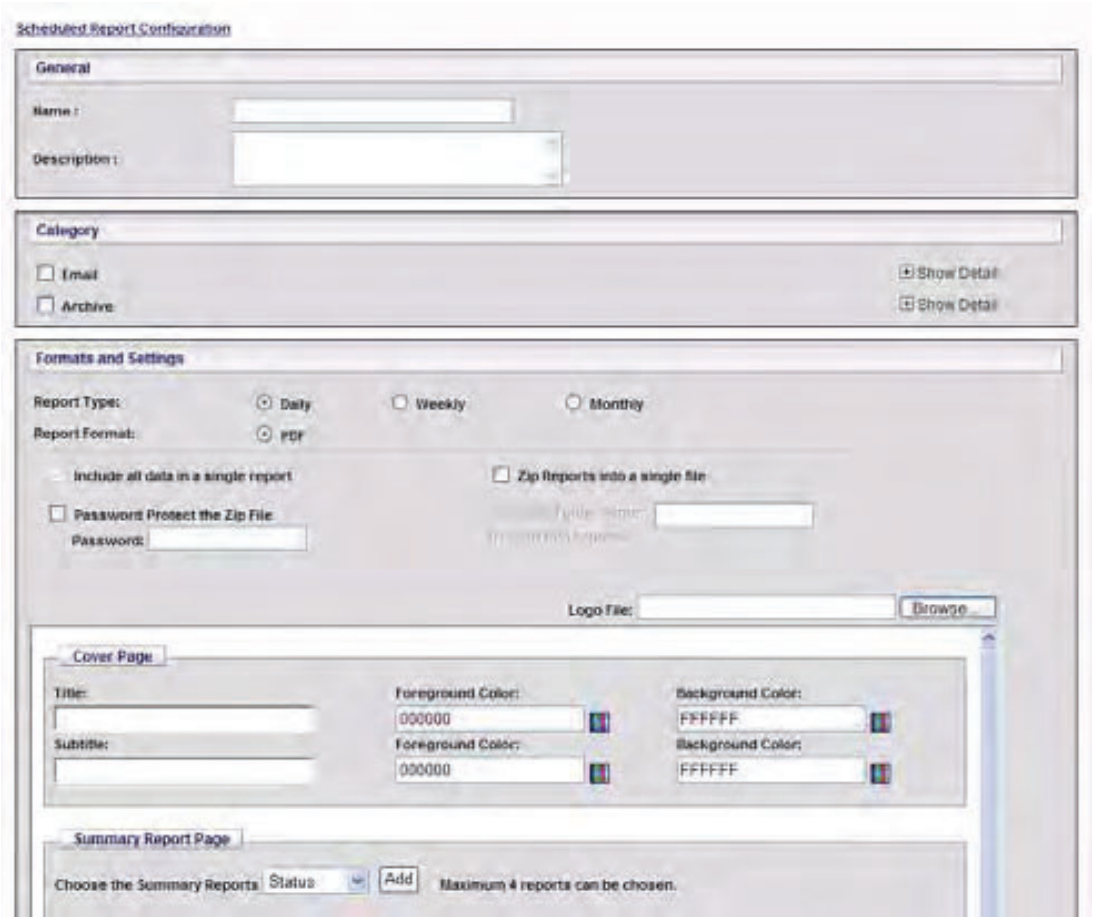
To configure SSL-VPN scheduled reports and summarization, perform the following tasks:

1. On the **SSL-VPN** tab, navigate to **Configuration > Scheduled Reports**.
2. Click the **Add** button.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1541
Keysight, EX-1045
Page 1541

3. The Scheduled Report Configuration form displays. Fill out the fields accordingly. For more information, see the following sections:
   - “Configuring Scheduled Reports” on page 134
   - “Scheduling PDF Compliance Reports” on page 144



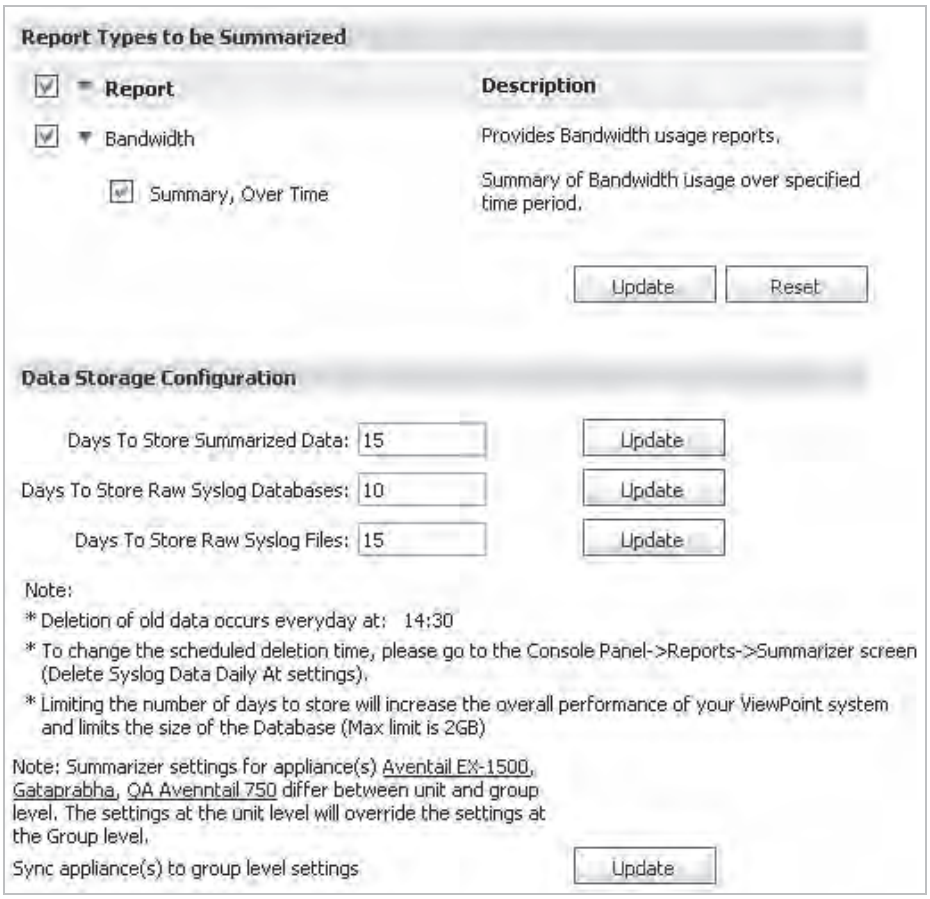

# Configuring SSL-VPN Summarization

1. On the **SSL-VPN** tab, navigate to **Configuration > Summarizer Settings**. The reports that can be summarized for a SSL-VPN appliance are configurable at either global or unit level. The screen displays the configuration appropriate for the level. The report type lists can also be expanded for a detailed description of report content.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1542
Keysight, EX-1045
Page 1542

The report types you can summarize are shown below.




SSL-VPN reports generated in ViewPoint can be exported in PDF format, providing easy online transfer. For more information about the Summarizer and exporting reports in PDF format, see:

- “Selecting Reports for Summarization” on page 137
- “Configuring Data Storage Settings” on page 139
- “Using Summarize Now” on page 76
- “Scheduling PDF Compliance Reports” on page 144



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1543
Keysight, EX-1045
Page 1543

# CHAPTER 17
# Viewing SSL-VPN Reports

This chapter describes the available reports for SonicWALL SSL-VPN appliances.

For information on how to configure scheduled reports and summarization, see:

- “Using and Configuring SSL-VPN Reporting” on page 295

Select from the following reports:

- “Viewing General Status Reports” section on page 299
- “Viewing SSL-VPN Bandwidth Reports” section on page 301
- “Using SSL-VPN Custom Reports” section on page 307
- “Viewing SSL-VPN Resources Reports” section on page 325
- “Viewing SSL-VPN Authentication Reports” section on page 330
- “Viewing the SSL-VPN Log” section on page 332

# Viewing General Status Reports

The **General > Status** page contains information about the SSL-VPN appliance or group of SSL-VPN appliances.

To view the Status page, perform the following steps:

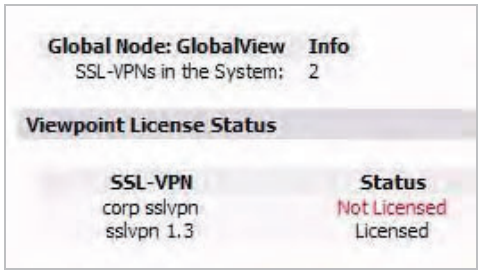
1. Click the **SSL-VPN** tab.
2. Select **MyReportsView** or an SSL-VPN appliance in the left pane.



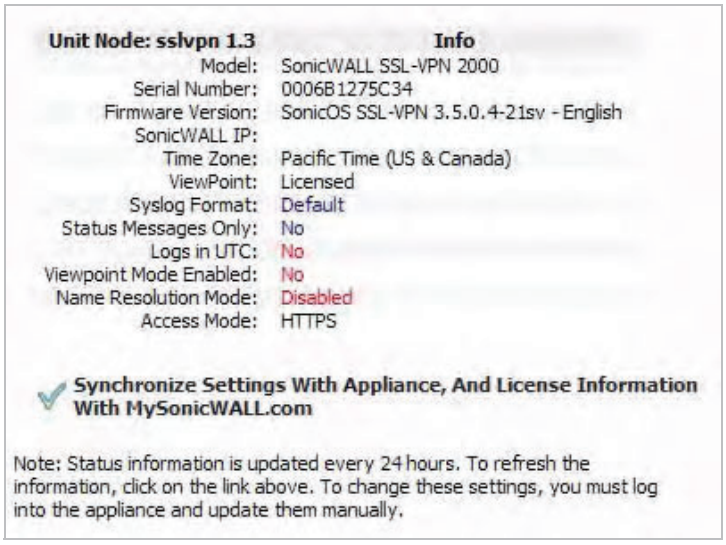
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1544
Keysight, EX-1045
Page 1544

3. In the center pane, expand the **General** tree and click **Status**. The Status page displays.

   When **MyReportsView** is selected, the Status page displays the license status of all SSL-VPN appliances.




   When a unit is selected, the Status page displays information about the SSL-VPN appliance, including model, serial number, firmware version, time zone, license status, log settings, and other settings.




4. In the unit view, to synchronize settings with the SSL-VPN appliance and license information with MySonicWALL, click **SynchronizeSettings With Appliance, And License Information With Mysonicwall.com**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1545
Keysight, EX-1045
Page 1545

# Viewing SSL-VPN Bandwidth Reports

Bandwidth reports display the amount of data transferred through one or more selected SSL-VPN appliances.

Bandwidth reports are an ideal starting point for viewing overall bandwidth usage. You can view bandwidth usage view by hour, day, or over a period of days. Additionally, you can view the top users of bandwidth.

From this information, you can determine network strategies. For example, if you need more bandwidth, you might need to upgrade network equipment, or you might simply need to curtail the bandwidth usage of a few employees.



**Note** All reports appear in the time zone of the selected appliance.

Select from the following:


- "Viewing SSL-VPN Bandwidth Summary Reports" on page 301
- "Viewing SSL-VPN Top Users of Bandwidth Reports" on page 303
- "Viewing SSL-VPN Bandwidth Usage Over Time Reports" on page 304
- "Viewing SSL-VPN Top Users of Bandwidth Over Time Reports" on page 306


# Viewing SSL-VPN Bandwidth Summary Reports

The Bandwidth Summary report shows the number of connections handled by a SSL-VPN appliance during each hour of the specified day, or at the global level, by each SSL-VPN appliance for the day.

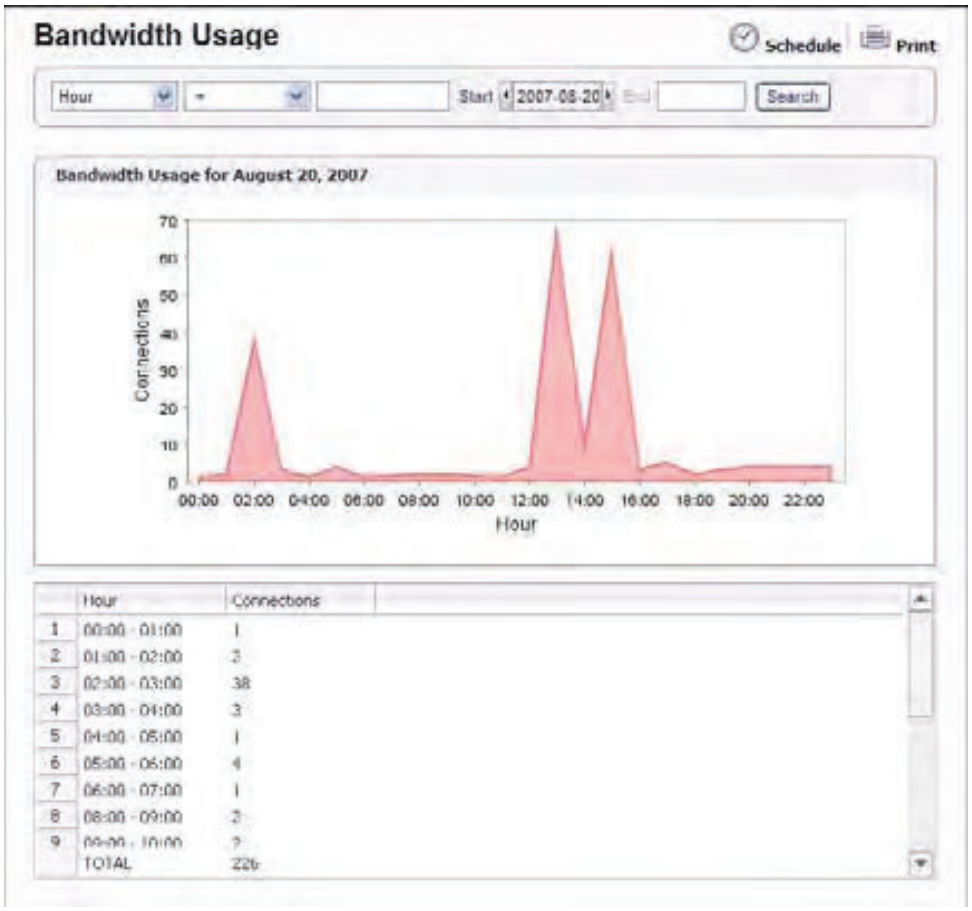
To view the Bandwidth Summary report, perform the following steps:

1. Click the **SSL-VPN** tab.
2. Select the global icon or a SSL-VPN appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1546
Keysight, EX-1045
Page 1546

3. Expand the **Bandwidth** tree and click **Summary**. The Summary page displays.




4. The graph displays the number of connections to the SSL-VPN appliance during each hour of the day.
5. The table contains the following information:
   - **Hour**—when the sample was taken.
   - **Connections**—number of connections to the SSL-VPN appliance
6. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report, click the **Start** field to access the drop-down calendar.
7. After selecting a date, click **Search**. The ViewPoint Reporting Module displays the report for the selected day.

**Note** The date setting will stay in effect for all similar reports during your active login session.



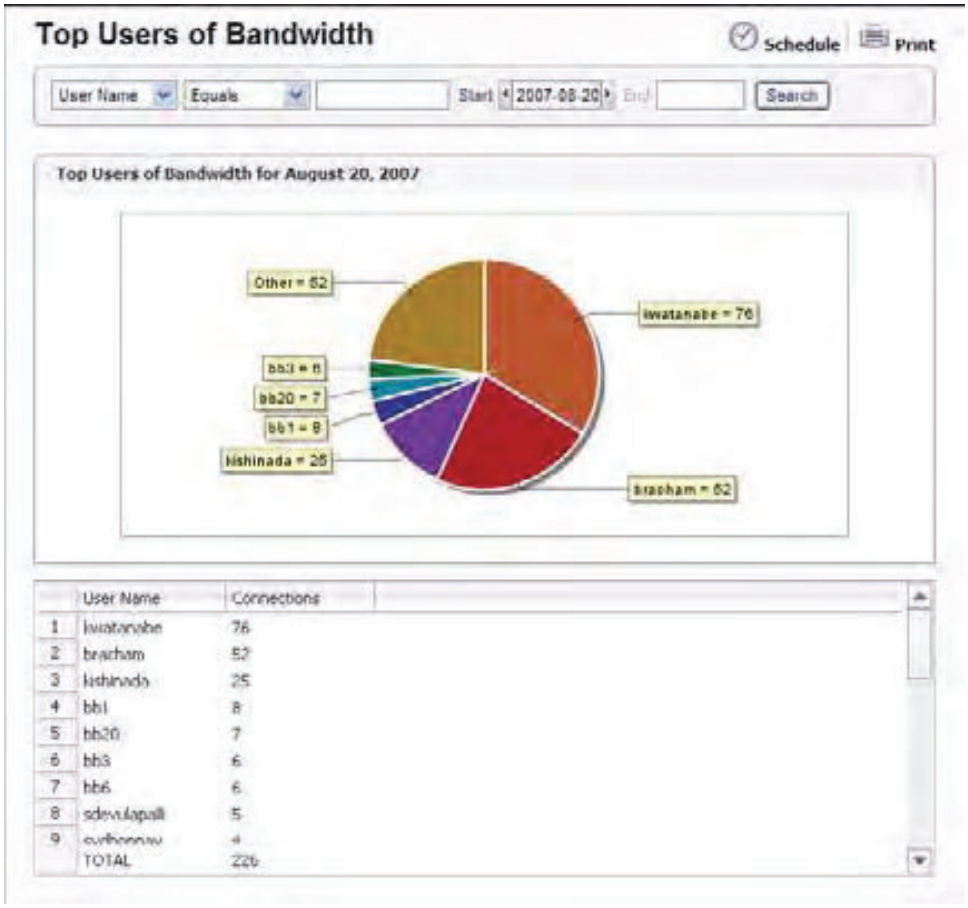
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1547
Keysight, EX-1045
Page 1547

# Viewing SSL-VPN Top Users of Bandwidth Reports

The Top Users report displays the users who used the most connections on the specified date.

To view the Top Users report, perform the following steps:

1. Click the **SSL-VPN** tab.
2. Select a SSL-VPN appliance.
3. Expand the **Bandwidth** tree and click **Top Users**. The Top Users page displays.




4. The pie chart displays the percentage of connections used by each user.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1548
Keysight, EX-1045
Page 1548

5. The table contains the following information for all users:
   - **Users**—the user name
   - **Connections**—number of connection events or “hits”
6. By default, the ViewPoint Reporting Module shows yesterday’s report, a pie chart for the top six users, and a table for all users. To change the date of the report, click the **Start** field to access the drop-down calendar.
7. To display a limited number of users, use the Search Bar fields.



**Note** The search bar fields use pattern matching with operators such as “contains”. For example, “john” will match john_smith, john42, or big_john.

8. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected day.



**Note** The date setting will stay in effect for all similar reports during your active login session.

# Viewing SSL-VPN Bandwidth Usage Over Time Reports

The Bandwidth Usage Over Time report displays the daily number of connections handled by a SSL-VPN appliance or a group of SSL-VPN appliances for the specified time period.

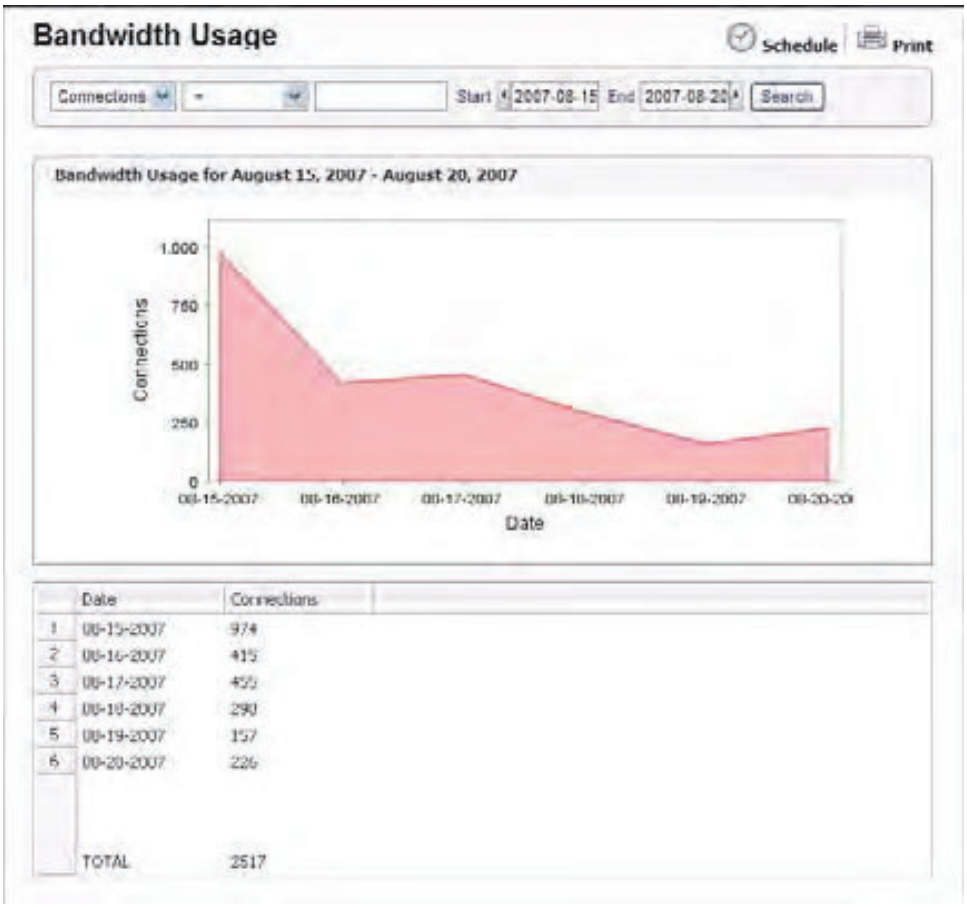
To view the Bandwidth Usage Over Time report, perform the following steps:

1. Click the **SSL-VPN** tab.
2. Select the global icon or a SSL-VPN appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1549
Keysight, EX-1045
Page 1549

3. Expand the **Bandwidth** tree and click **Over Time**. The Over Time page displays.




4. The graph displays the number of connections during each day of the specified time period.
5. The table contains the following information:
   - **Date**—when the sample was taken
   - **Connections**—number of hits
6. To change the date of the report, use the Search Bar and click the **Start** or **End** fields to access the drop-down calendar.
7. When you are finished, click **Search.** The ViewPoint Reporting Module displays the report for the selected date range.

**Note** These date settings will stay in effect for all similar reports during your active login session.



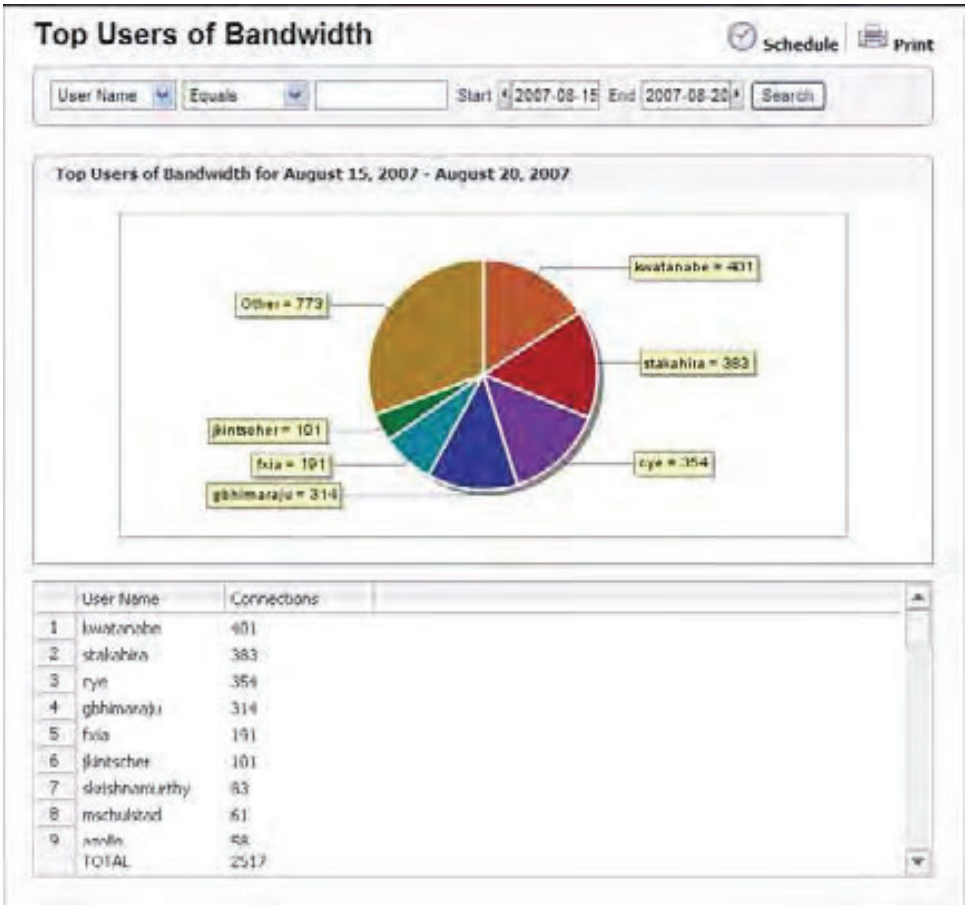
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1550
Keysight, EX-1045
Page 1550

# Viewing SSL-VPN Top Users of Bandwidth Over Time Reports

The Top Users Over Time report displays the users who used the most connections during the specified date range. This report is available at the unit level.

To view the Top Users Over Time report, perform the following steps:

1. Click the **SSL-VPN** tab.
2. Select a SSL-VPN appliance.
3. Expand the **Bandwidth** tree and click **Top Users Over Time**. The Top Users Over Time page displays.




4. The pie chart displays the percentage of connections used by the top users.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1551
Keysight, EX-1045
Page 1551

5. The table contains the following information for all users:
   - **Users**—the user name of the user
   - **Connections**—number of connection events or “hits”
6. The ViewPoint Reporting Module shows yesterday’s report. To change the date range of the report, click the **Start** or **End** field to access the drop-down calendar.
7. To display a limited group of users, enter the user IDs in the **Search Bar** fields.



**Note** The search bar fields use pattern matching with operators such as “contains”. For example, “john” will match john_smith, john42, or big_john.

8. When you are finished, click **Search.** The ViewPoint Reporting Module displays the report for the selected users and date range.



**Note** These settings will stay in effect for all similar reports during your active login session.

# Using SSL-VPN Custom Reports

Custom Reports are available at the unit level for appliances visible on the SSL-VPN tab. Log Viewer must be enabled for the appliance. For information about enabling Log Viewer, see “Viewing the SSL-VPN Log” on page 332.

When configuring a Custom Report on the Resource Activity page, the Template Section acts as a query builder. You select the criteria for the report that you want, and SonicWALL ViewPoint uses your input to query the raw syslog database for the information, and then outputs the report. The Template Section consists of two parts: the Date/Time section and the Report Layout section.

After building your query in the Template Section and clicking the **Generate Report** button, the report is displayed in the Report Section. The Report Section is displayed in the lower half of the page, under the Template Section; this layout is called Split Mode. You can easily toggle between Split Mode and Full Mode. Full Mode can be used to display only the Template Section or only the Report Section in a full page view.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1552
Keysight, EX-1045
Page 1552

The Report Section displays the report and provides controls for pagination, printing, and exporting the report in PDF or CSV format. You can also click the **Save Template** button in this section if you want to save the settings for this report as a template for reuse later. See the following sections for detailed information:


- "Toggling Between Split Mode and Full Mode" on page 308
- "Configuring the Date and Time for Custom Reports" on page 311
- "Configuring the Report Layout and Generating the Report" on page 314
- "Generating the Custom Report" on page 320
- "Viewing a Custom Report" on page 321
- "Printing a Page or Exporting the Report as a PDF or CSV File" on page 323
- "Saving the Report Template" on page 324

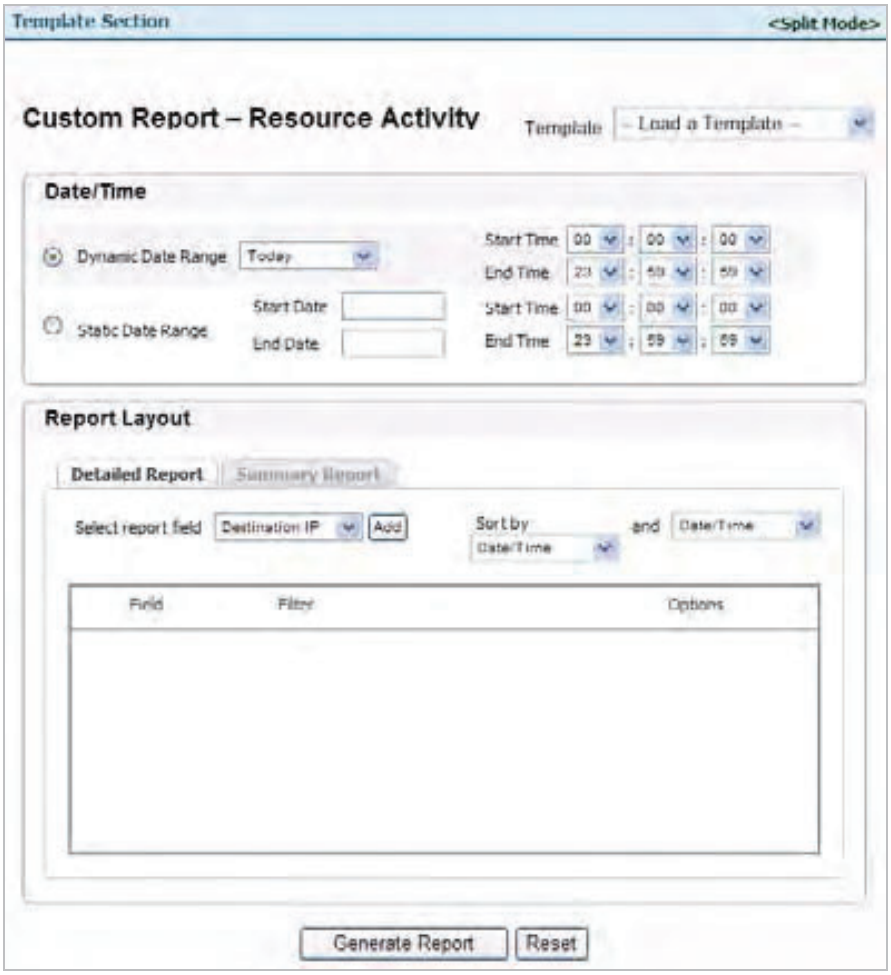

# Toggling Between Split Mode and Full Mode

The Custom Report page contains two main sections, the Template Section and Report Section, which can be displayed together or independently depending on the mode.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1553
Keysight, EX-1045
Page 1553

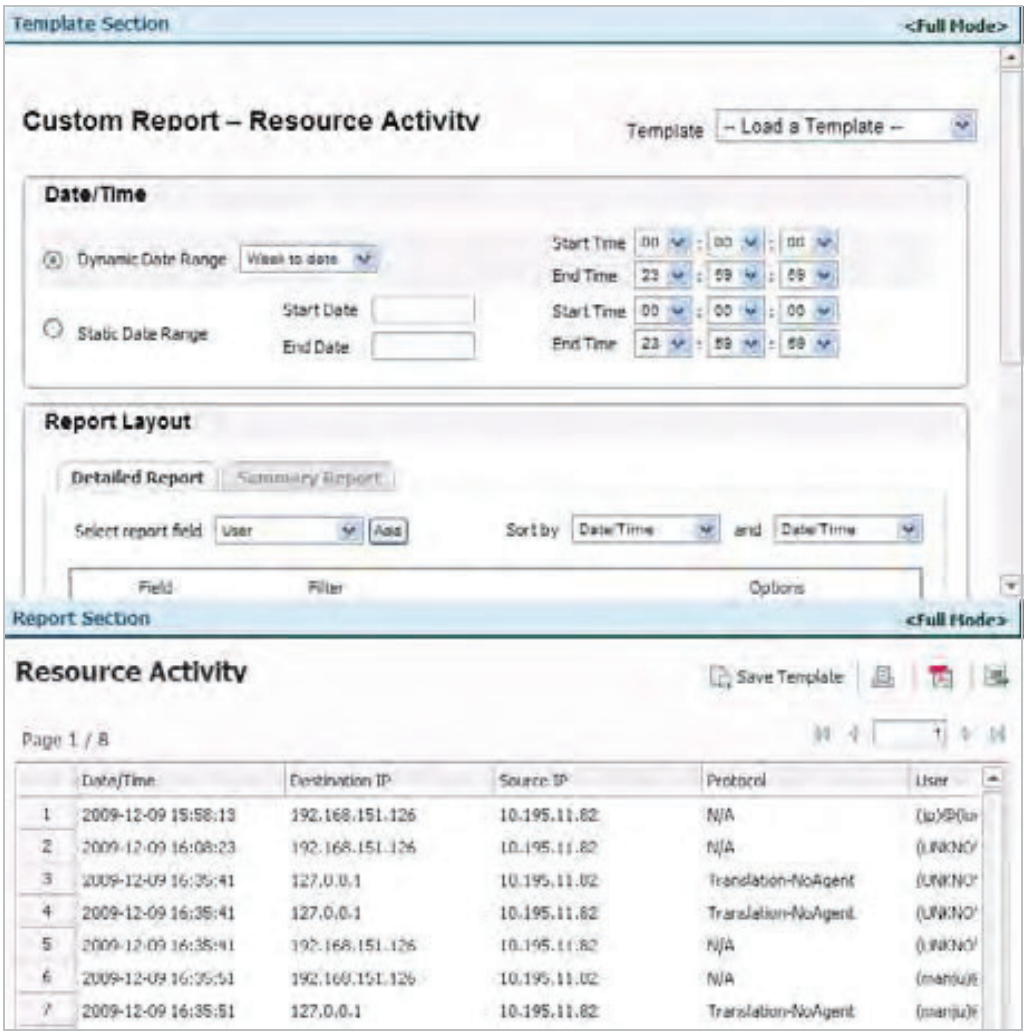
When the Custom Report page is initially displayed for a selected appliance, the Template Section is displayed in Full Mode. Split Mode is available, but the Report Section displays no data until a report has been generated. The image below shows the Custom Report > Resource Activity page with the Template Section displayed in Full Mode.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1554
Keysight, EX-1045
Page 1554

After generating a report, the page automatically changes to Split Mode and displays the report settings in the Template Section in the top half of the page and the report results in the Report Section in the lower portion. The image below shows the Template Section and Report Section displayed in Split Mode.




At any time, you can change to Full Mode if you want to display either the Template Section or the Report Section individually. From Full Mode, you can easily change back to Split Mode.

To toggle between Split Mode and Full Mode:

1. Select a unit for which Log Viewer is enabled, and then navigate to the **Custom Report** page.
2. On a page that is currently displayed in Full Mode, to change the view to Split Mode click the **<Split Mode>** button at the right side of the section heading.



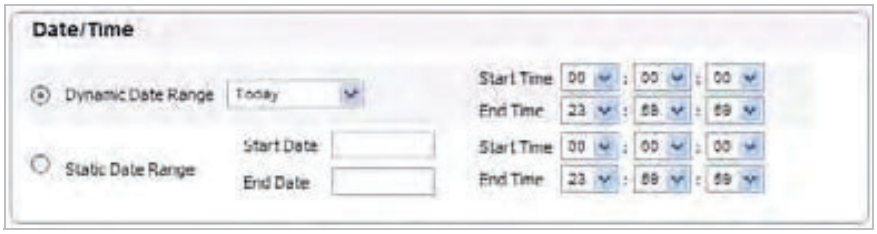
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1555
Keysight, EX-1045
Page 1555

3. On a page that is currently displayed in Split Mode, do one of the following to change to a Full Mode display of either the Template Section or the Report Section:
   - Click the **<Full Mode>** button to the right of the Template Section heading.
   - Click the **<Full Mode>** button to the right of the Report Section heading.

# Configuring the Date and Time for Custom Reports

At the top of the Template Section of the Custom Report page, the Date/Time region provides a way to designate the time period to use when generating the report. You can select either a Dynamic Date Range or a Static Date Range.

Both the Dynamic Date Range and the Static Date Range provide Start Time and End Time settings. By using the Start Time and End Time fields, you can specify the exact hour, minute, and second for both the beginning and the end of the period for the report. When a start and end time is specified for a date range containing multiple days, the start/end times are applied to each day of the period when analyzing data for the report. The default is to include data for the full 24 hours in each day of the date range.




## Dynamic Date Range

The **Dynamic Date Range** selection allows you to select from four date ranges and to specify the exact starting and ending times on the days in the selected date range for the log data to be used for the report.

For the **Dynamic Date Range**, you can select from the following four date choices:

- **Today** – Uses log data from the current date, beginning just after midnight
- **Yesterday** – Uses log data from just after midnight of the previous day, up to and including the most recent log message from the current date



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1556
Keysight, EX-1045
Page 1556

- **Week to Date** – Uses log data from the current date, plus the seven preceding days
- **Month to Date** – Uses log data from the same date as the current date in the previous month, up to and including the most recent log message from the current date

When generating a report with a template containing a dynamic date range setting, the dates used when referencing the log data are relative to the current date. Thus, two reports generated from the same template on different days will provide different results.

To select a Dynamic Date Range:

1. Select a unit for which Log Viewer is enabled, and then navigate to the page under **Custom Report** for the report type you want.
2. In the Template Section under Date/Time, select the **Dynamic Date Range** radio button.
3. In the drop-down list, select **Today**, **Yesterday**, **Week to Date**, or **Month to Date**.
4. For the **Start Time**, select the hour, minute, and second from the drop-down lists in the Dynamic Date Range row. These settings specify the earliest data to be included in the report, for each day of the date range.
5. For the **End Time**, select the hour, minute, and second from the drop-down lists. These settings specify the most recent data to be included in the report, for each day of the date range.
6. To change the settings back to the defaults, click **Reset** at the bottom of the Template Section. Note that this will change the Report Layout region as well as the Date/Time region back to default settings.

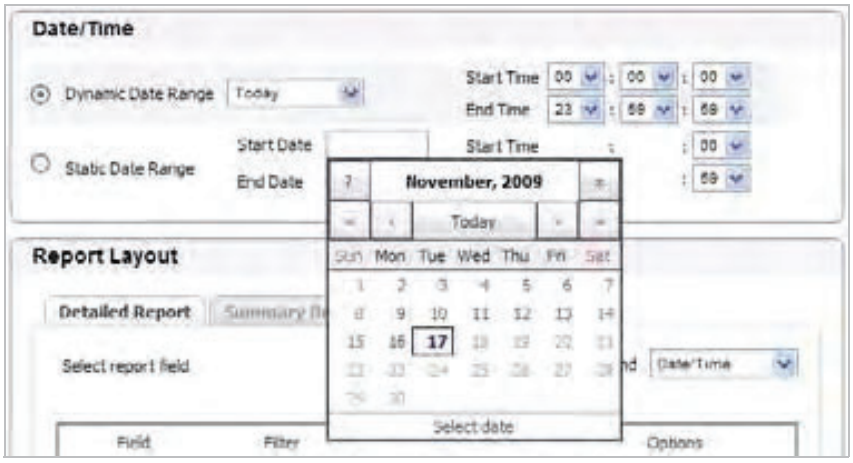
## Static Date Range

The **Static Date Range** selection allows you to specify the exact dates, starting, and ending times on the days in the selected date range for the log data to be used for the report. You can specify a single date or a date range, and indicate the exact hour, minute, and second for both the beginning and the end of the daily period for the report.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1557
Keysight, EX-1045
Page 1557

A popup calendar makes it easy to select the Start Date and End Date for the date range, as shown below.




To specify a Static Date Range:

1. Select a unit for which Log Viewer is enabled, and then navigate to the page under **Custom Report** for the report type you want.
2. In the Template Section under Date/Time, select the **Static Date Range** radio button.
3. Click the **Start Date** field to access the pop-up calendar.
4. Use the navigation arrows near the top of the calendar to change the year or month. Click the **<<** button to move to the previous year, or hold the button to select from a list of years. Click the **>>** button to move to the next year, or hold the button to select from a list of years. Similarly, click the **<** or **>** to move back or ahead by one month, or hold the button to select from a list of months.
5. Click the desired start date in the calendar. This adds the date to the **Start Date** field and closes the calendar.
6. Click the **End Date** field to access the pop-up calendar.
7. Use the navigation arrows near the top of the calendar to change the year or month.
8. Click the desired end date in the calendar. This adds the date to the **End Date** field and closes the calendar.
9. For the **Start Time**, select the hour, minute, and second from the drop-down lists in the Static Date Range row. These settings specify the earliest data for each day in the date range to be included in the report.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1558
Keysight, EX-1045
Page 1558

10. For the **End Time**, select the hour, minute, and second from the drop-down lists. These settings specify the most recent data for each day in the date range to be included in the report.
11. To change the settings back to the defaults, click **Reset** at the bottom of the Template Section. Note that this will change the Report Layout region as well as the Date/Time region back to default settings.

# Configuring the Report Layout and Generating the Report

Located in the Template Section of the Custom Report page below the Date/Time region, the Report Layout region provides a way to specify the type of data to include, and the format of the report. The Report Layout region has a Detailed Report tab and a Summary Report tab. The report appearance and the way information is organized is quite different between a Detailed Report and a Summary Report.

The Detailed Report tab contains a list of data categories that you can add as report fields, and allows you to specify query values for each. The categories you select will appear as column headings in the report.

The Summary Report tab allows you to structure a report showing the top elements of Resource Activity. You can select the number of top elements, what to base the comparisons on, and the two data categories to evaluate when determining the top elements. The generated report provides graphical output that you can click to drill down for detailed information.

For more information about each of these Report Layout tabs, see the following sections:

- “Detailed Reports” on page 315
- “Summary Reports” on page 318

For information about the Filter operators, see the following section:

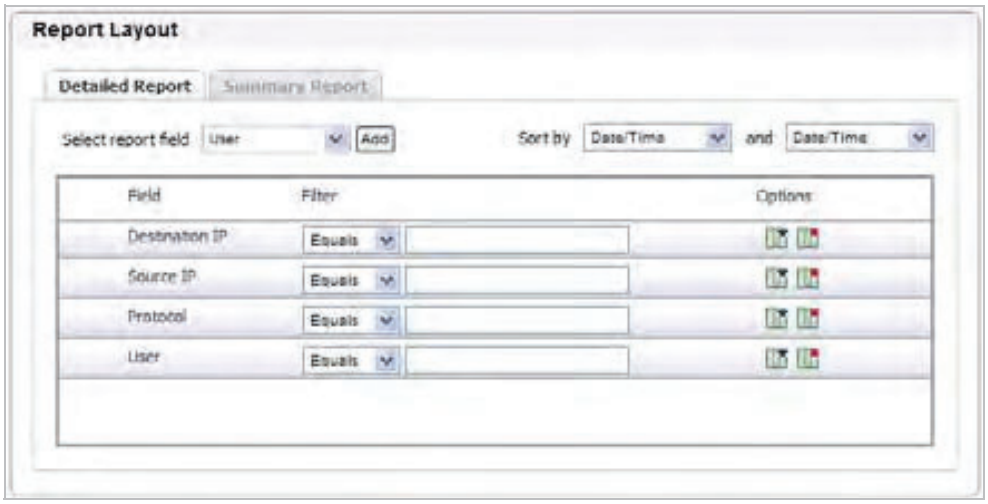
- “Filter Operators” on page 319



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1559
Keysight, EX-1045
Page 1559

# Detailed Reports

The Detailed Report tab is the default view in the Report Layout region.




For a SSL-VPN Resource Activity report, the **Select report field** drop-down list contains four data categories that you can add as column headings in the report. The categories are:

- **Destination IP** – Adds a column containing the IP address of each accessed resource
- **Protocol** – Adds a column containing the protocol used by the traffic
- **Source IP** – Adds a column containing the IP address of each system which accessed a resource
- **User** – Adds a column containing the user ID

To include a field in the report, select a choice from the list and then click **Add**. When you click **Add**, a row is populated in the table below, which has three column headings: Field, Filter, and Options.

**Note** When you place your mouse cursor over the row, under the Field heading, the cursor changes to a “move” cursor. You can drag and drop the rows to rearrange the column ordering in the final report.

In the Filter column, two fields are displayed: an operator field and an input field. The operator field is a drop-down list containing the operator choices for the selected report field. See “Filter Operators” on page 319 for a description of each operator. The input field can be a drop-down list or a standard input field, depending on the selected report field.

The operators and input fields are defined in Table 7 for each report field.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1560
Keysight, EX-1045
Page 1560

*Table 7 Operators and Input Fields for Each Data Type*

| Data Type | Operators | Input Field |
|---|---|---|
| Destination IP | Equals<br>Starts with<br>Ends with<br>Contains | The input field is a standard input field where you can type in the numbers to match, such as 192 or 10.25. Leave the input field blank if you choose not to filter by a certain destination IP address. |
| Protocol | Equals<br>Start with<br>End with<br>Contains | The input field is a standard input field where you can type in the protocol to match, such as FTP. Leave the input field blank if you choose not to filter by a certain protocol. |
| Source IP | Equals<br>Starts with<br>Ends with<br>Contains | The input field is a standard input field where you can type in the numbers to match, such as 192 or 10.25. Leave the input field blank if you choose not to filter by a certain source IP address. |
| User | Equals<br>Start with<br>End with<br>Contains | The input field is a standard input field where you can type in the user ID to match. Leave the input field blank if you choose not to filter by a certain user. |

In the Options column, two icons are displayed: an Eye  and an X  . You can click the Eye to toggle whether the report field on that row will be displayed in the final report. This allows you to filter the report results based on the selected report field and related filter value, but not display the field as a column. When you click on the Eye icon within a row, the eye closes  to show that this field will not be displayed in the final report. The filter value will still be used to filter results from the raw syslog database to apply towards the report.

For example, you might specify the following Field/Operator/Filter Value: **Protocol**/**=**/**http**. It would make sense to click the Eye icon to disable the Protocol field from being shown in the report, since it would always just be “http” and would not add any interesting information to the final report. Contrast this with simply specifying the Protocol field and leaving the Filter Value blank, in which case you would want to enable the Eye so that this column would appear in the report showing a variety of protocols such as udp/dns, tcp/http, udp/ntp, or numbered protocols such as udp/389 (the LDAP protocol) or tcp/445 (MS Server Message Block (SMB) file sharing).

Clicking the X icon under Options deletes the selected report field from the table, so it will not be used to generate the report results nor will it be displayed in the report. Use the X icon instead of the Eye when you do not choose to filter the report results based on the field.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1561
Keysight, EX-1045
Page 1561

The Detailed Report tab also contains the Sort By drop-down list. The list contains the Date/Time option and any other report fields that you have selected from the eight data types. The choice you select will be used to order the results in the report from the first page to the last. The selection in the left drop-down list is used for the first sorting, then the selection in the right drop-down list is used to sort and group the entries within each group resulting from the the first sorting.

To configure a detailed report:

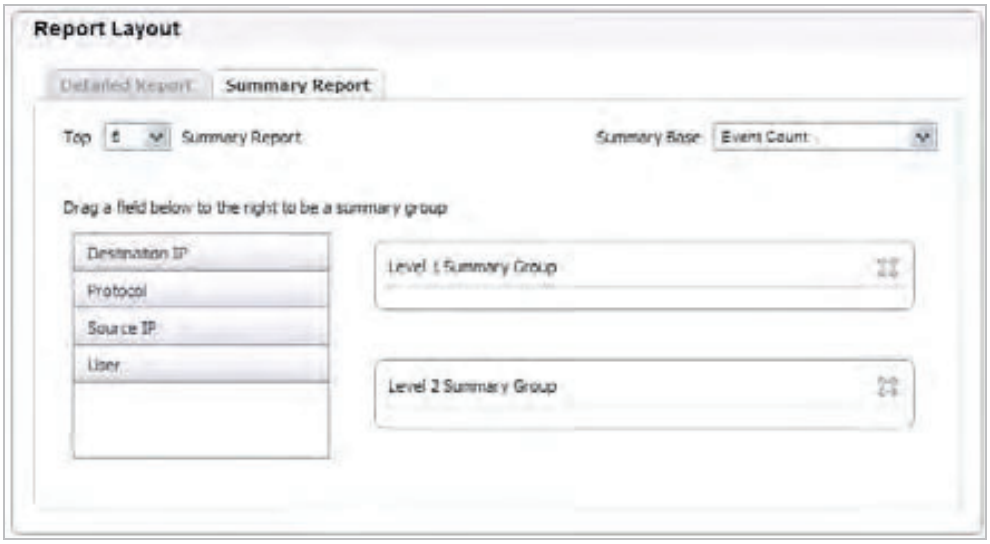
1. Select a unit for which Log Viewer is enabled, and then navigate to the page under **Custom Report** for the report type you want.
2. In Report Layout region of the Template Section of the Custom Report page, select the **Detailed Report** tab.
3. In the **Select report field** drop-down list, select a data type to include in the report, and then click **Add**. A row for this field is populated in the table below. Repeat this step to add other fields.
4. Optionally select an operator from the drop-down list under **Filter** in a table row, and type in or select an input value to be matched when the database is queried. Repeat this step for other rows to add filter values for those fields.
5. To prevent a field from appearing in the final report, click the **Eye** icon in that row so that the eye appears closed. To allow the field to be displayed in the report, click the closed **Eye** icon to return it to normal appearance.
6. To delete a field from the table, click the **X** icon in that row.
7. To sort the report pages by a different field than the default of Date/Time, select the desired field from the **Sort by** drop-down list.
8. To change the settings back to the defaults, click **Reset** at the bottom of the Template Section. Note that this will change the Date/Time region and the Report Layout region back to default settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1562
Keysight, EX-1045
Page 1562

# Summary Reports

The Summary Report tab is available in the Report Layout region of the Template Section.




The Top drop-down list provides selections for the number of entries to display in the report. For example, if the User field is selected below as a Summary Group, and 5 is selected in the Top drop-down list, the report will provide entries for the top five users. For all Custom Reports, available numbers in the Top drop-down list are 5, 10, 20, 50, and 100.

The Summary Base drop-down list offers a selection of traffic types that will be used to determine the top usage for the selected field. For a SSL-VPN Resource Activity report, the only Summary Base choice is **Event Count**.

Below the Top and Summary Base fields, you can create one or two Summary Groups from the choices listed on the left side. For a SSL-VPN Resource Activity report, the choices are **Destination IP**, **Protocol**, **Source IP**, or **User**.

To select a field for a Summary Group, simply drag and drop the desired field from the list to either the Level 1 Summary Group or Level 2 Summary Group boxes. When the field name is dragged to one of these, the operator drop-down list and filter input value field are displayed, allowing you to specify values to match when the data is searched. See “Filter Operators” on page 319 for a description of each operator.

Either the Level 1 Summary Group field or the Level 2 Summary Group field can be used alone; the resulting report will look the same in both cases.

When both the Level 1 and Level 2 Summary Group fields are populated, the report will display the top entries for the Level 2 field for each of the top entries for the Level 1 field. For example, if User is dragged to the Level 1 Summary



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1563
Keysight, EX-1045
Page 1563

Group and Domain is dragged to the Level 2 Summary Group, and 5 is selected in the Top drop-down list, the generated report will display the top five domains visited by each of the top five users.

To configure a summary report:

1. Select a unit for which Log Viewer is enabled, and then navigate to the page under **Custom Report** for the report type you want.
2. In Report Layout region of the Template Section of the Custom Report page, select the **Summary Report** tab.
3. In the **Top** drop-down list, select the number of entries to be displayed in the report.
4. In the **Summary Base** drop-down list, use the default, **Event Count**.
5. To specify the field for the **Level 1 Summary Group**, click and drag the desired field from the list on the left to the **Level 1 Summary Group** field, and then release your mouse button to drop the field into position. The filter operator and input field are displayed next to the field name.




6. To specify the field for the **Level 2 Summary Group**, click and drag the desired field from the list on the left to the **Level 2 Summary Group** field, then release your mouse button to drop the field into position. The filter operator and input field are displayed next to the field name.
7. To specify a filter operator and filter value for a Summary Group, select the operator from the drop-down list next to the field and type a filter value into the input field to the right of the operator.
8. To change the settings back to the defaults, click **Reset** at the bottom of the Template Section. Note that this will change the Date/Time region as well as the Report Layout region back to default settings.

## Filter Operators

When configuring the Report Layout on either the Detailed Report tab or the Summary Report tab, you can specify filter values to be matched in the database during report generation. Depending on the selected field type, text string or numeric, several filter operators are available. The filter operators are used with a filter input value to determine which data should be included in the report.

The operators are defined as shown in Table 8.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1564
Keysight, EX-1045
Page 1564

*Table 8 Filter Operators*

| Operator | Definition |
|---|---|
| Equals | Only data that exactly matches the filter input text will be included in the report |
| Start with | Data that begins with the input text will be included in the report |
| End with | Data that ends with the input text will be included in the report |
| Contains | Data that contains the input text will be included in the report |
| = | Only data that exactly matches the filter input numerical value will be included in the report |
| > | Data values that are greater than the input numerical value will be included in the report |
| >= | Data values that are greater than or equal to the input numerical value will be included in the report |
| <= | Data values that are less than or equal to the input numerical value will be included in the report |
| < | Data values that are less than the input numerical value will be included in the report |
| != | Data values that are not equal to the input numerical value will be included in the report |

# Generating the Custom Report

The Generate Report button at the bottom of the Template Section is used to create the report. Before clicking Generate Report, use the Template Section to specify the time period for the report and the contents and layout of the report.

**Note** Custom Reports are available at the unit level and Log Viewer must be enabled for the appliance. For information about enabling Log Viewer, see “Viewing the SSL-VPN Log” on page 332.

To generate a custom report:

1. Select a unit for which Log Viewer is enabled, and then navigate to the page under **Custom Report** for the report you want.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1565
Keysight, EX-1045
Page 1565

2. In the **Date/Time** region of the Template Section, specify the time period that the report will cover. For detailed information and instructions, see “Configuring the Date and Time for Custom Reports” on page 311.
3. In the **Report Layout** region of the Template Section, specify the contents and appearance of the report. For detailed information and instructions, see “Configuring the Report Layout and Generating the Report” on page 314.
4. Click **Generate Report** to create the report using the specified configuration.

# Viewing a Custom Report

After you click Generate Report, the Report Section is displayed in Split Mode in the lower half of the main window, even if you previously were in Full Mode for the Template Section.

Pagination controls are displayed at the upper right of the report, just below the Save Template button and the printer, PDF, and Excel icons. Navigation buttons are provided to take you to the first page, next page, previous page, and last page, or you can specify an exact page number in the field.

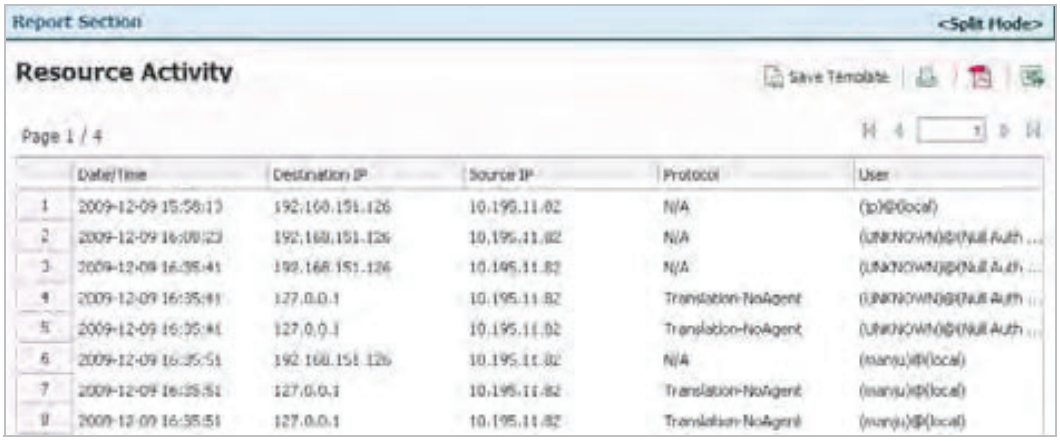



In a Detailed Report, the selected report fields are displayed as column headings. You can click on any column heading to sort that page by the values in the column that you click. Click again to toggle between ascending and descending order on that page. When you navigate away from that page and

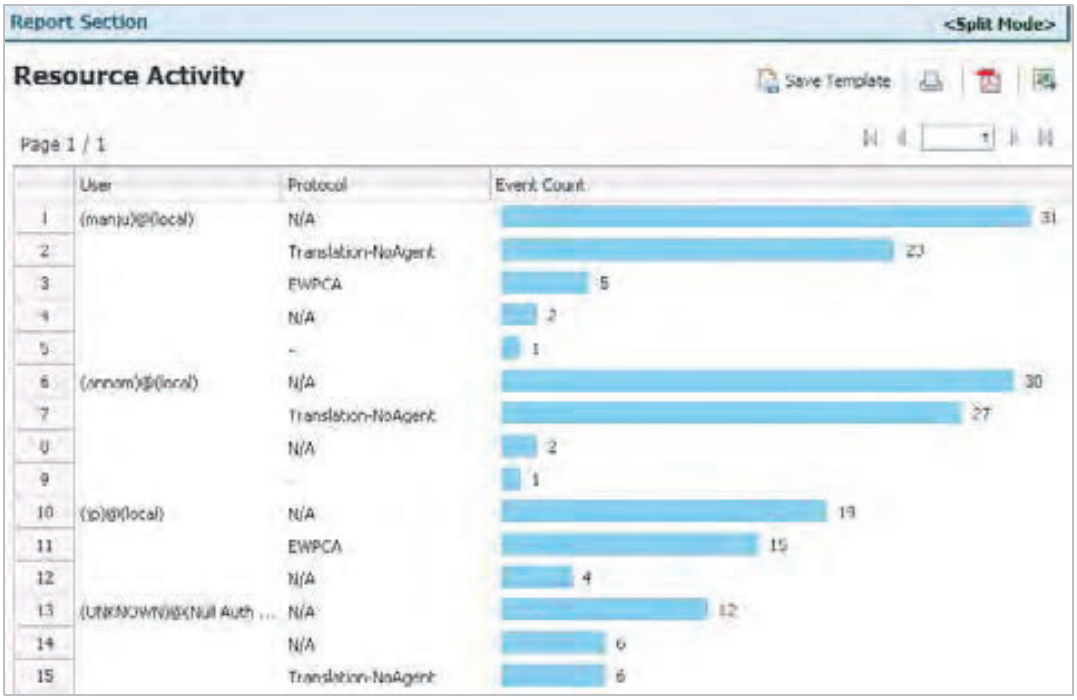


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1566
Keysight, EX-1045
Page 1566

then come back using the pagination controls, the page reverts to the original sorting order as specified in the **Sort by** field of the Template Section before generating the report.




In a Summary Report, the Report Section displays the event count as horizontal bar charts. This lets you see the information at a glance, such as who had the most resource activity and which protocols they used the most.



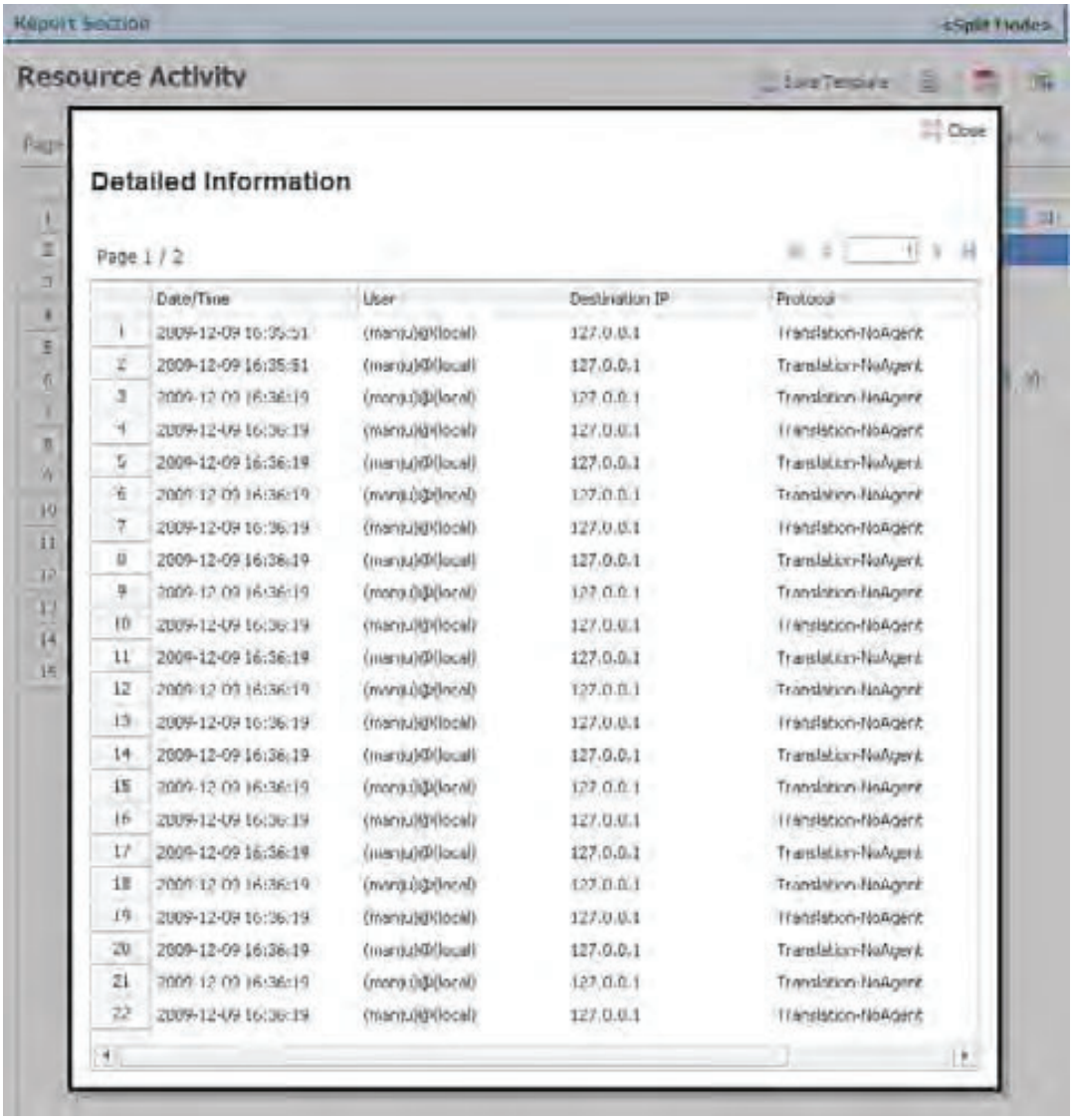

You can click on a bar in the chart to pop up detailed information, just like the detailed report with all of the columns for all fields. The report lists details about this Summary Group field only. For example, if the Summary Group contains the User field and you click on a bar for one of the top users, the report displays the date and time of all resource activity for the user, and



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1567
Keysight, EX-1045
Page 1567

includes data for every field available for detailed reports. A scroll bar is provided along the bottom of the Detailed Information window to allow viewing of all four fields plus the date and time column.

The Detailed Information window is shown below.




# Printing a Page or Exporting the Report as a PDF or CSV File

To print the current page of the report, click the printer icon at the top of the Report Section. Your normal print dialog box pops up. This prints only the page that is currently displayed.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1568
Keysight, EX-1045
Page 1568

To export the entire report in PDF format, click the PDF icon at the top of the Report Section. A PDF file is generated showing the report results in table format.

To export the entire report in Microsoft Excel Comma Separated Value (CSV) format, click the Excel icon at the top of the Report Section. A CSV file is generated showing the report results in spreadsheet format.

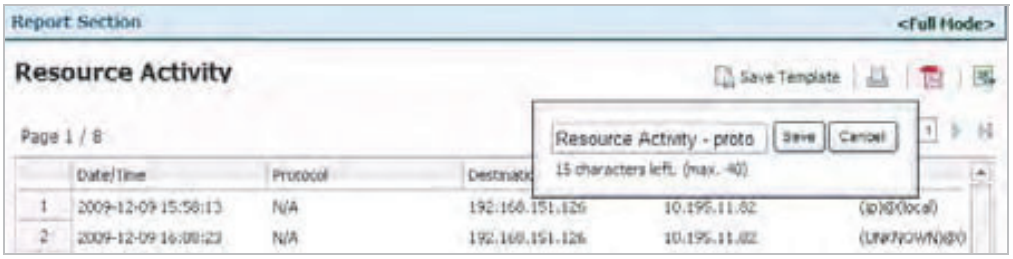
The PDF can contain a maximum of 10,000 records. If your report contains more than 10,000 records, you can use the Static Date Range fields to adjust the dates and regenerate the report to shorten its length. You can save the PDF or CSV file using any filename and location.

# Saving the Report Template

After generating the report, you can save the settings for this report as a template for reuse. You can select the saved template from the Template Section at a later time, and use it to generate a report using the same settings. The template is saved for the currently selected appliance and for the specific user. The saved template will not be available for other appliances or for other users.

To save the report template:

1. In the Report Section in the upper right corner, click the **Save Template** button.




2. In the popup dialog box, type in a descriptive name for the template, up to 40 characters. The number of remaining characters allowed in the name is displayed below the input field and changes as you type.
3. Click **Save**. If you are in a Full Mode display of the Report Section, you can verify that the template has been saved by changing back to Split Mode and viewing the contents of the **Template** drop-down list.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1569
Keysight, EX-1045
Page 1569

# Viewing SSL-VPN Resources Reports

Resources reports provide information on the amount of data transmitted through the selected SSL-VPN appliance by each service or protocol.

Resources reports are useful for revealing inappropriate usage of bandwidth and can help determine network policies. For example, if there is a large spike of bandwidth usage, you can determine whether this is caused by regular Web access, someone using FTP to transfer large files, an attempted Denial of Service (DoS) attack, or another service.

**Note** All reports appear in the appliance's time zone.

The procedures for viewing the Resources Reports are described in the following sections:

- "Viewing SSL-VPN Resources Summary Reports" on page 325
- "Viewing SSL-VPN Resources Top Users Reports" on page 327

**Note** You cannot view resources reports from the global view.

## Viewing SSL-VPN Resources Summary Reports

The Resources Summary report displays the number of connections handled by each service or protocol during the specified day.

To view the Resources Summary report, perform the following steps:

1. Click the **SSL-VPN** tab.
2. Select a SSL-VPN appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1570
Keysight, EX-1045
Page 1570

3. Expand the **Resources** tree and click **Summary**. The Resources Summary page displays.




4. The graph displays the number of connections used by each service or protocol during the day.
5. The table contains the following information:



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1571
Keysight, EX-1045
Page 1571

- **Resource name**—the service or protocol
- **Connections**—number of connection events or “hits”

6. To view the user detail for a particular resource, click the resource slice in the pie chart or the resource name in the table to drill down for this information.




7. To return to the Resources > Summary page, click the **Go Back** button.
8. To change the date of the report, use the Search Bar and click the **Start** field to access the drop-down calendar.
9. When you are finished, click **Search.** The ViewPoint Reporting Module displays the report for the selected date.

**Note** This date setting will stay in effect for all similar reports during your active login session.

# Viewing SSL-VPN Resources Top Users Reports

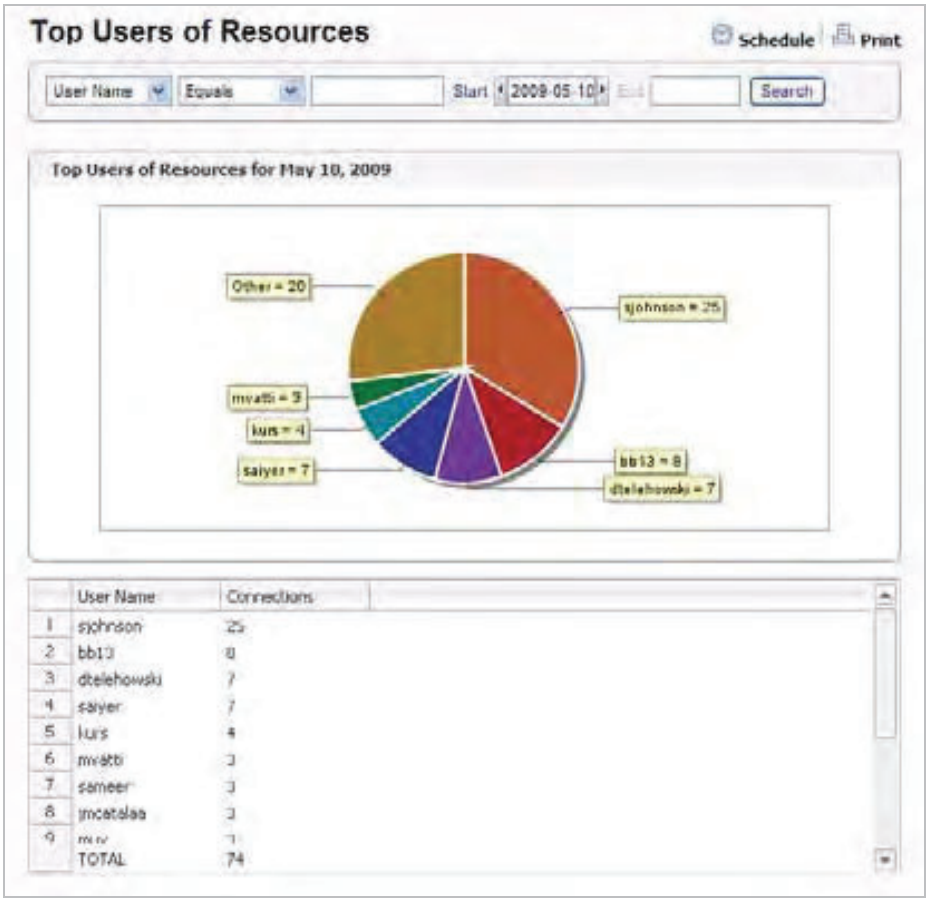
The Resources Top Users report displays the users who used the most connections on the specified date.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1572
Keysight, EX-1045
Page 1572

To view the Resources Top Users report, perform the following steps:

1. Click the **SSL-VPN** tab.
2. Select a SSL-VPN appliance.
3. Expand the **Resources** tree and click **Top Users**. The Top Users page displays.

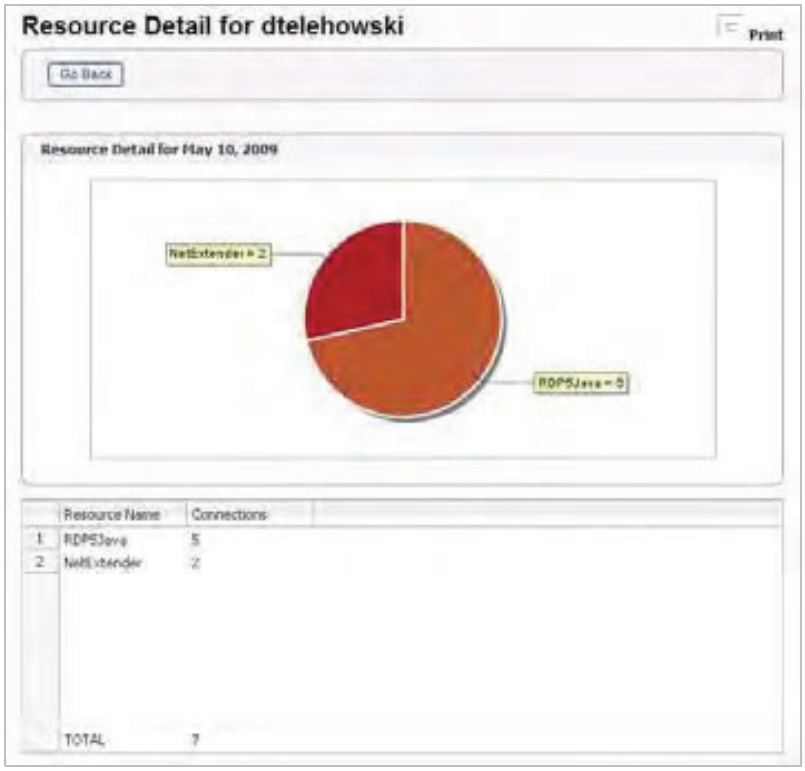



4. The pie chart displays the percentage of connections used by each user.
5. The table contains the following information for all users:
   - **Users**—the user name
   - **Connections**—number of connection events or “hits”



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1573
Keysight, EX-1045
Page 1573

6. To view the resources by service or protocol used by a particular user, click the user slice in the pie chart or the user name in the table to drill down for this information.




7. To return to the Resources > Top Users page, click the **Go Back** button.
8. By default, the ViewPoint Reporting Module shows yesterday's report, a pie chart for the top six users, and a table for all users. To change the date of the report, click the **Start** field to access the drop-down calendar.
9. To display a limited number of users, use the Search Bar fields.

**Note** The search bar fields use pattern matching with operators such as "contains". For example, "john" will match john_smith, john42, or big_john.

10. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected day.

**Note** The date setting will stay in effect for all similar reports during your active login session.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1574
Keysight, EX-1045
Page 1574

# Viewing SSL-VPN Authentication Reports

The Authentication reports show user logins and failed login attempts. Authentication reports are available at the unit level.

**Note** All reports appear in the appliance's time zone.

Select from the following:

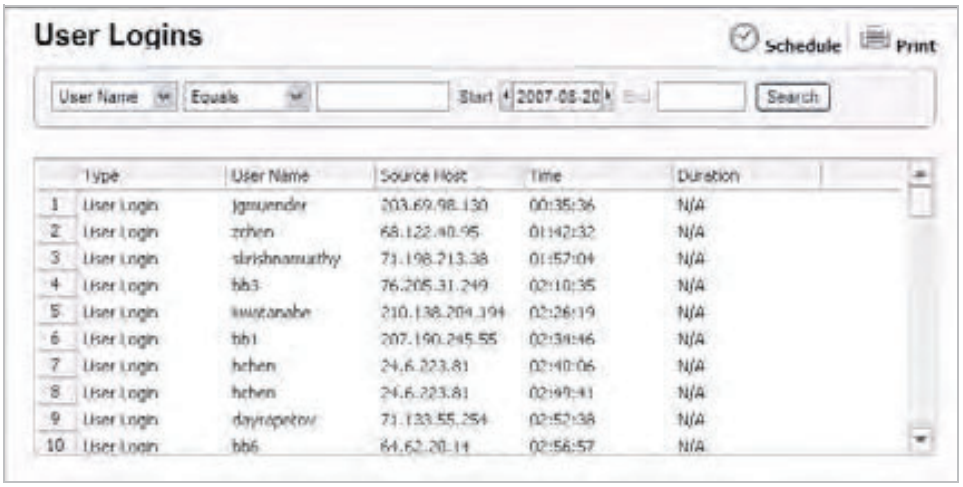

- "Viewing SSL-VPN User Login Reports" on page 330
- "Viewing SSL-VPN Failed Login Reports" on page 331


## Viewing SSL-VPN User Login Reports

The user login report shows the user name, source host IP address, and time of login for users that logged on to the SSL-VPN appliance during the specified day.

To view the User Login report, perform the following steps:

1. Click the **SSL-VPN** tab.
2. Select a SSL-VPN appliance.
3. Expand the **Authentication** tree and click **User Login**. The User Login page displays.




4. The table contains the following information:
   - **Type**—equal to User Login
   - **User Name**—the user name



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1575
Keysight, EX-1045
Page 1575

- **Source Host**—the IP address of the user's computer
- **Time**—the time that the user logged in
- **Duration**—the duration of the user login session

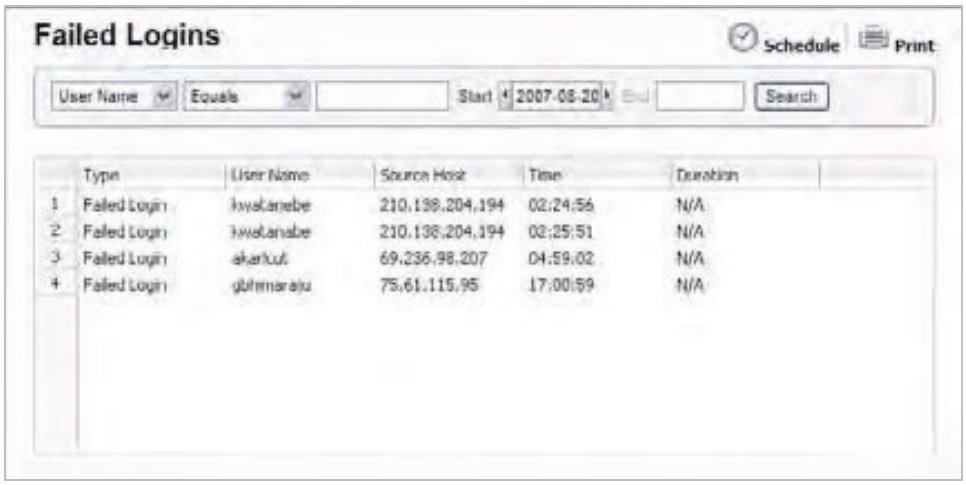
5. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report, use the Search Bar and click the **Start** field to access the drop-down calendar.
6. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing SSL-VPN Failed Login Reports

The failed login report shows failed login attempts for users who attempted to log into the SSL-VPN appliance during the specified day. This report is useful for identifying unauthorized access attempts and potentially malicious activity.

To view the Failed Login report, perform the following steps:

1. Click the **SSL-VPN** tab.
2. Select a SSL-VPN appliance.
3. Expand the **Authentication** tree and click **Failed Login**. The Failed Logins page displays.




4. The table contains the following information:
   - **Type**—equal to Failed Login
   - **User Name**—the user name
   - **Source Host**—the IP address of the user's computer
   - **Time**—the time that the user attempted to log in



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1576
Keysight, EX-1045
Page 1576

- **Duration**—not applicable

5. The ViewPoint Reporting Module shows yesterday's report. To change the date of the report, use the Search Bar and click the **Start** field to access the drop-down calendar.
6. When you are finished, click **Search**. The ViewPoint Reporting Module displays the report for the selected date.

# Viewing the SSL-VPN Log

The Log Viewer contains detailed information on each transaction that occurred on the SSL-VPN appliance. This information is stored for the time that you specified in the configuration settings.



**Note** The Log Viewer displays raw log information for every connection. Depending on the amount of traffic, this can quickly consume a large amount of space in the database. It is highly recommended to be careful when choosing the number of days of information that will be stored. For more information, see "Scheduling and Configuring Reports" on page 133.

## Viewing the Log for a SSL-VPN Appliance

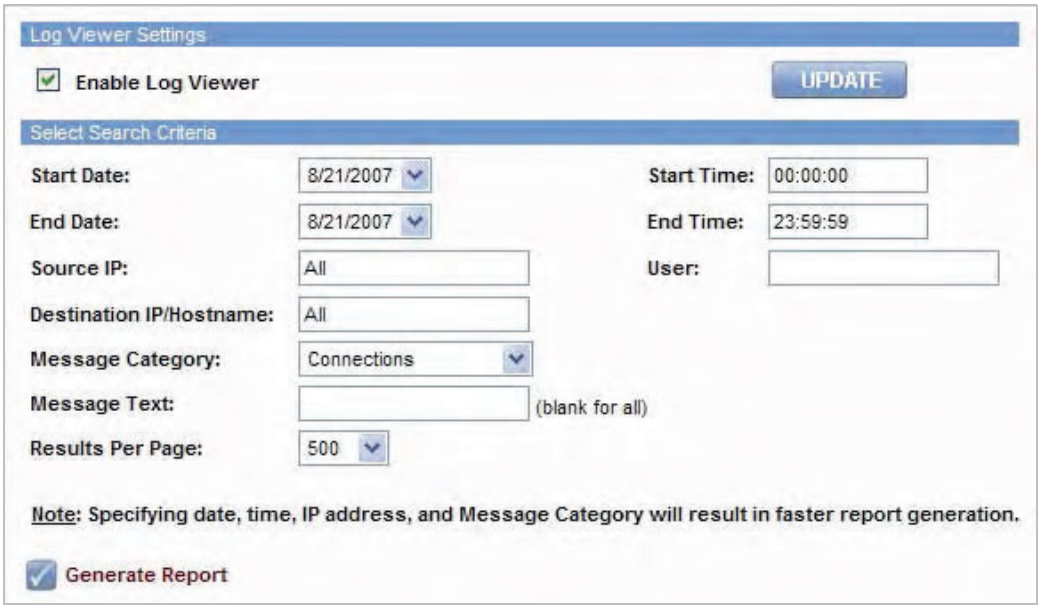
To view the Log, perform the following steps:

1. Click the **SSL-VPN** tab.
2. Select a SSL-VPN appliance.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1577
Keysight, EX-1045
Page 1577

3. Expand the **Log Viewer** tree and click **Search**. The Search page displays.




4. Select **Enable Log Viewer** and then click **Update** to turn on collection of raw data in the database and enable viewing of that log data. This can consume a large amount of space in your database. Review your database space constraints before enabling the log viewer.
5. Under Select Search Criteria, select the date range to view data from in the **Start Date** and **End Date** fields.
6. Enter the starting time of events to view in the **Start Time** field.
7. Enter the ending time of events to view in the **End Time** field.
8. To limit the report to data originating from specific IP addresses, enter the source IP address in the **Source IP** field. To view all IP addresses, enter **All**.
9. To view log entries for data originating from a particular user, enter the user name in the **User** field.
10. To limit the report to data going to specific IP addresses or hosts, enter the destination IP address or host name in the **Destination IP/Hostname** field. To view data for all IP addresses, enter **All**.
11. Select the type of events to view from the **Message Category** list box. You can select from the following:
    - All Categories
    - Connections
    - Rejected Connections



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1578
Keysight, EX-1045
Page 1578

- User Events
- Unrecognized Events

12. To limit the report to messages containing a specific text string, enter the text in the **Message Text** field. Leave the field blank to view all messages.
13. Select the number of entries to display per page from the **Results Per Page** field.
14. Click **Generate Report**. The Log Search Results page displays.




15. To view the next page of entries, click **Next**.
16. To generate another report, click **Search** again in the **Log Viewer** tree.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1579
Keysight, EX-1045
Page 1579

# Appendix A

# Installing SonicWALL ViewPoint

This appendix is designed to help you install SonicWALL ViewPoint. If you have not used SonicWALL ViewPoint before, you might want to familiarize yourself with SonicWALL ViewPoint concepts and features. This appendix contains the following sections:


- "About Installing and Upgrading SonicWALL ViewPoint" section on page 336
- "Activating SonicWALL ViewPoint on Your Appliances" section on page 340
- "Installing Universal Management Suite" section on page 342
- "Upgrading SonicWALL ViewPoint 5.1 to 6.0" section on page 349
- "Registering SonicWALL ViewPoint" section on page 351
- "Configuring Deployment Settings" section on page 354
- "Upgrading from ViewPoint to GMS" section on page 356
- "Miscellaneous Procedures and Troubleshooting Tips" section on page 368





CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1580
Keysight, EX-1045
Page 1580

# About Installing and Upgrading SonicWALL ViewPoint

You can either perform a fresh installation of SonicWALL ViewPoint using the installer or upgrade a previous installation of SonicWALL ViewPoint, patched or unpatched.

The upgrade installer checks with the SonicWALL backend to see if the SonicWALL ViewPoint deployment has valid support. If it does not, then the upgrade discontinues. When the SonicWALL ViewPoint installer detects that the SonicWALL backend site is not accessible, it prompts the user to enter an Upgrade Key.




If the key is valid, it allows the upgrade to continue. If the key is invalid, the installation fails.

**Note** The upgrade key can be obtained by contacting SonicWALL Technical Support.

# Installing SonicWALL ViewPoint

This chapter describes how to install or upgrade SonicWALL ViewPoint.

To install or upgrade SonicWALL ViewPoint, complete the following procedures:

- Review the installation requirements. See "Installation Overview" on page 336.
- To install SonicWALL ViewPoint, see "Installing Universal Management Suite" on page 342.

# Installation Overview

The SonicWALL ViewPoint Installation program is an HTML-launched installer that automatically detects whether you are installing on Windows Server 2000/2003/2008. After the installation program detects the operating system, the installation procedure is identical.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1581
Keysight, EX-1045
Page 1581

# System Requirements

**Note** SonicWALL does not support installations of ViewPoint running on any virtualization software, such as VMware.

Before installing SonicWALL ViewPoint, review the requirements in the following sections:


- Operating System Requirements, page 337
- Database Requirements, page 337
- Java Requirements, page 338
- Browser Requirements, page 338
- Hardware Requirements, page 339
- SonicWALL Appliance and Firmware Support, page 339
- Network Requirements, page 339
- MySonicWALL Account Requirements, page 340


## Operating System Requirements

In order to install and run SonicWALL ViewPoint, you must be logged in as the administrator. SonicWALL ViewPoint is supported on the following operating systems:

- Windows Server 2008 SBS, 64-bit
- Windows Server 2008 Standard (SP1), 32-bit and 64-bit
- Windows Server 2003 (SP2), 32-bit and 64-bit
- Windows Server 2000 (SP4)
- Windows 7, 32-bit and 64-bit
- Windows Vista (SP1), 32-bit and 64-bit
- Windows XP Professional (SP3), 32-bit

In all instances, SonicWALL ViewPoint runs as a 32-bit application.

## Database Requirements

For fresh installations or after upgrading from 5.1, SonicWALL ViewPoint 6.0 supports the following database:

- MySQL 32-bit version 5.0.83 for Windows, bundled with SonicWALL ViewPoint 5.1 and above



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1582
Keysight, EX-1045
Page 1582

The MySQL 5.0 separate installer that was provided with SonicWALL ViewPoint 5.0 is still supported.

The requirements for the MySQL server are as follows:

- Windows 2000 (SP4) and newer Windows operating systems
- Minimum 300 GB hard disk space
- Minimum 2 GB RAM
- NTFS file system
- Not a Virtual Machine (VM)

After upgrading from 5.1, SonicWALL ViewPoint 6.0 supports the following databases only when the database was already in use prior to upgrading:

- Microsoft SQL Server 2005 (SP2), 32-bit and 64-bit, as follows:
  - SQL Server 2005 Workgroup
  - SQL Server 2005 Standard
  - SQL Server 2005 Enterprise
  - SonicWALL ViewPoint does not support Microsoft SQL 2005 Express
- Microsoft SQL Server 2000 (SP4)
- Microsoft Desktop Engine (MSDE) bundled with ViewPoint

## Java Requirements

Java Plug-in version 1.6 or higher is required on client machines when accessing the SonicWALL ViewPoint application interface. SonicWALL Universal Management Suite (UMS) automatically downloads the latest Java Plug-in. SonicWALL UMS services use JRE 1.6. For the Web server, SonicWALL UMS uses Tomcat 6.0.20.

## Browser Requirements

- Microsoft Internet Explorer 6.0 or higher
- Mozilla Firefox 2.0 or higher
- Pop-up blocker disabled

SonicWALL ViewPoint supports SSL 3.0 / TLS 1.0 for HTTPS direct login to SonicWALL appliances from SonicWALL ViewPoint. For enhanced security across a SonicWALL ViewPoint network for installations that must comply with stringent regulatory compliance and account management controls as found in such standards as PCI, SOX, or HIPAA, the following browsers have SSL 3.0/TLS 1.0 as standard encryption protocols:

- Microsoft Internet Explorer 7.0 or higher



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1583
Keysight, EX-1045
Page 1583

- Mozilla Firefox 2.0 or higher

## Hardware Requirements

The hardware platform where SonicWALL ViewPoint is installed must meet the following requirements:

- x86 environment
- 3 GHz or faster single-CPU Intel processor
- Minimum 2 GB RAM
- At least 100 GB of free disk space



**Note** Ensure that the drive where SonicWALL ViewPoint is installed has ample space to store the SonicWALL ViewPoint log files.

SonicWALL ViewPoint requires large amounts of disk space for database storage. In early versions, the maximum raw syslog database size was 2 GB. SonicWALL ViewPoint now provides enhanced database capacity by creating a new 2 GB database everyday. Each file name includes the date it was created for easy reference.

## SonicWALL Appliance and Firmware Support

You can use SonicWALL ViewPoint reporting for the following SonicWALL security appliances:

- SonicWALL firewalls running SonicOS 1.0 or higher, or SonicWALL firmware 6.1.2.0 or higher
- SonicWALL SSL-VPN 200 / 2000 / 4000 running SonicOS SSL VPN 2.1 or higher
- SonicWALL SRA 4200 running SonicOS SSL VPN 3.5.0.11 or higher
- SonicWALL Aventail E-Class SRA EX-Series appliances running version 9.0 or higher
- SonicWALL CSM Series running SonicOS CF 1.0 or higher

## Network Requirements

To complete the SonicWALL ViewPoint deployment process, the following network requirements must be met:

### Syslog and SNMP Port Settings

You should either disable your personal firewall, or enable ports for syslog, syslog forwarding, and SNMP traps. The default syslog port is UDP 514 and the default SNMP port is UDP 162.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1584
Keysight, EX-1045
Page 1584

If the SonicWALL ViewPoint system is behind a gateway or firewall, you may need to open up these ports on that device.

### Static IP / DHCP

If accessed from the WAN interface, the SonicWALL appliance must have a static IP address. Otherwise, it may have either a static or dynamic IP address.

### HTTP / HTTPS

HTTP and HTTPS access for adding a SonicWALL appliance to ViewPoint is supported as follows:

- HTTP for access to a LAN IP address only
- HTTPS for access to a LAN IP or WAN IP address

## MySonicWALL Account Requirements

A MySonicWALL account is required to complete the SonicWALL UMS installation and registration process. If you do not already have a MySonicWALL account, open a Web browser and navigate to the following website:

http://www.mysonicwall.com

Follow the on-screen prompts to create a user account.

# Activating SonicWALL ViewPoint on Your Appliances

To use SonicWALL ViewPoint, you must license it on each SonicWALL security appliance for which you want reports. The SonicWALL appliance must be registered on MySonicWALL before you can purchase and activate the SonicWALL ViewPoint license for it. You must also enable the SonicWALL ViewPoint license on the appliance itself.

See the following sections:


- "Registering Your SonicWALL Appliance" on page 341
- "Activating the ViewPoint Software on Your Appliance" on page 341
- "Enabling the ViewPoint License on Your Appliance" on page 342

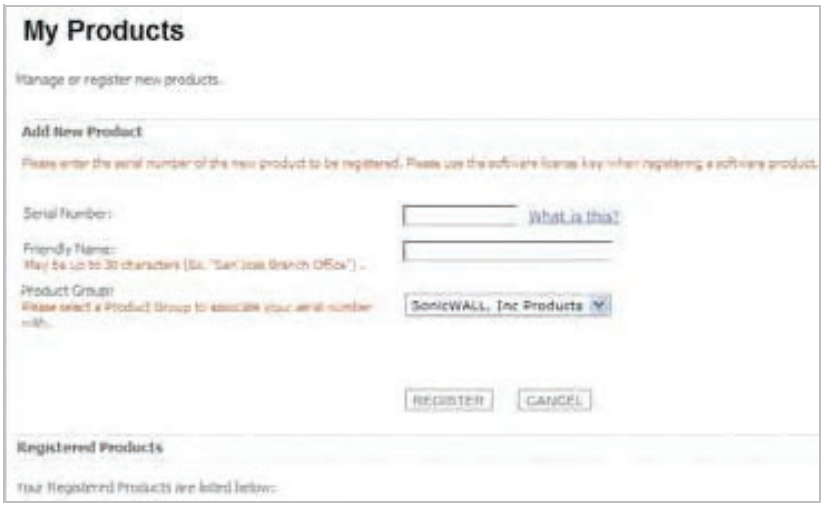



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1585
Keysight, EX-1045
Page 1585

# Registering Your SonicWALL Appliance

To register the SonicWALL appliance that ViewPoint will monitor, perform the following steps:

1. Log on to MySonicWALL.
2. Click **My Products**. The **SonicWALL My Products** page displays.




3. Enter your SonicWALL serial number in the **Serial Number** field.
4. Enter a descriptive name for the SonicWALL appliance in the **Friendly Name** field.
5. Select the **Product Group** from the drop-down list.
6. Click **Register**. The MySonicWALL website registers the SonicWALL appliance.

# Activating the ViewPoint Software on Your Appliance

To activate the SonicWALL ViewPoint software, perform the following steps:

1. Log on to mysonicwall.com.
2. Click the label of the newly registered SonicWALL appliance. The Service Management page displays.
3. Scroll down to locate the ViewPoint service and click **Enter Key**. The Activate Service page displays.
4. Enter the ViewPoint Activation Key in the **Activation Key** field. The ViewPoint Activation Key is printed on the ViewPoint Software License Certificate shipped with the ViewPoint package. If you purchased ViewPoint on mysonicwall.com, the key is emailed to you.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1586
Keysight, EX-1045
Page 1586

5. Click **Submit**. After the Activation Key is registered, a ViewPoint License Key will appear. Carefully write down the ViewPoint License Key in a safe place.

## Enabling the ViewPoint License on Your Appliance

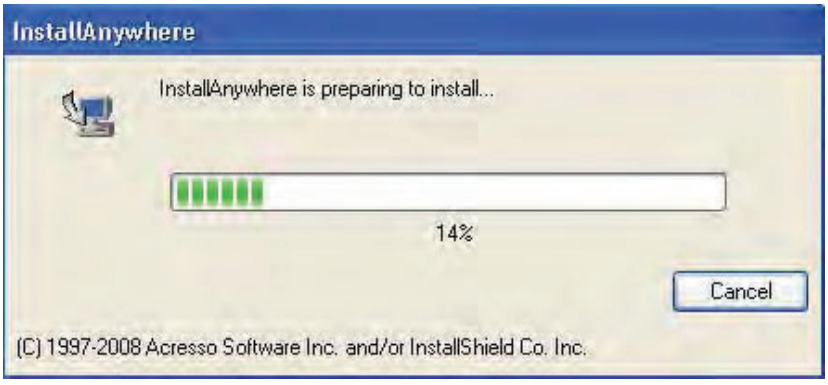
To enable the SonicWALL ViewPoint license, perform the following steps:

1. Log into the SonicWALL appliance.
2. Navigate to **Log > ViewPoint**. The **ViewPoint** page displays.
3. Enter the ViewPoint License Key provided by mysonicwall.com in the **Enter Upgrade Key** field.
4. Click **Apply**.
5. Restart the SonicWALL for the change to take effect.

# Installing Universal Management Suite

This section provides the procedures to install the SonicWALL Universal Management Suite (UMS) software. To install the SonicWALL UMS software, perform the following steps:

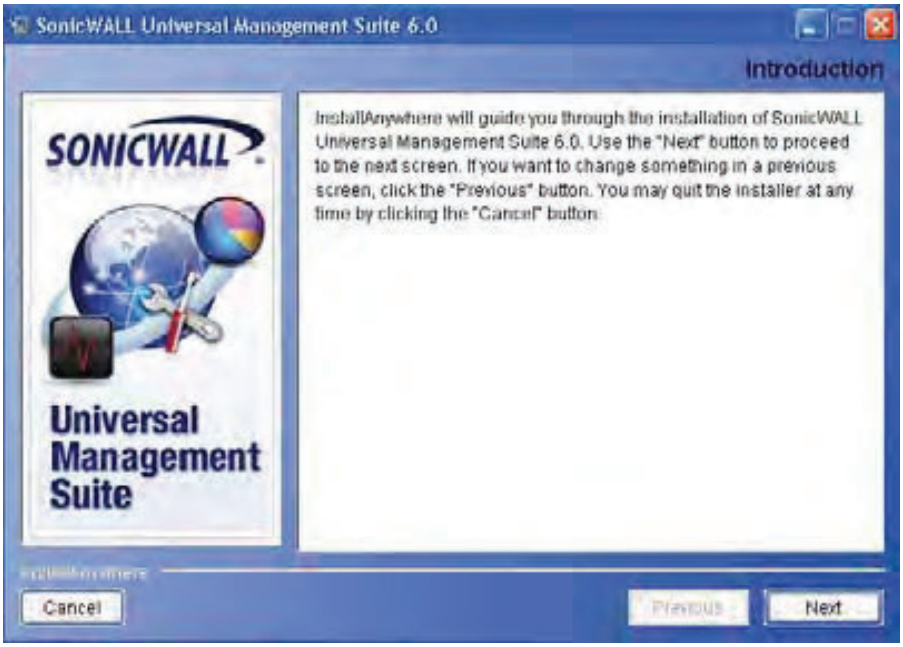
1. Log on to your SonicWALL ViewPoint management computer as **administrator** (Windows).
2. Run the SonicWALL ViewPoint installation file, **sw_gmsvp_win_eng_6.0.xxxx.xxxx.exe** (where "xxxx" represent the exact version numbers). It may take several seconds for the InstallAnywhere installer to initialize.



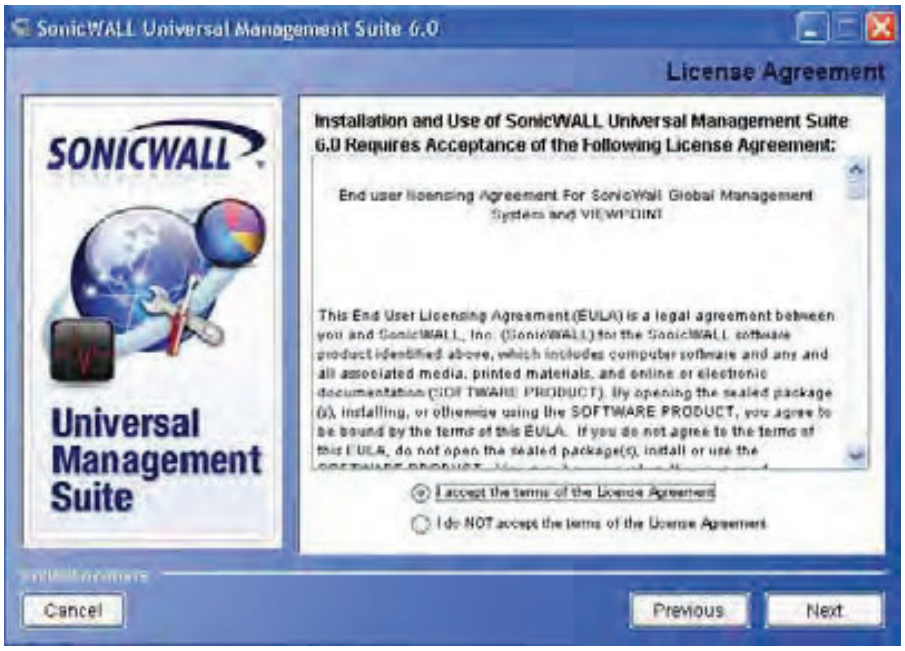



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1587
Keysight, EX-1045
Page 1587

3. In the Introduction screen, click **Next**.



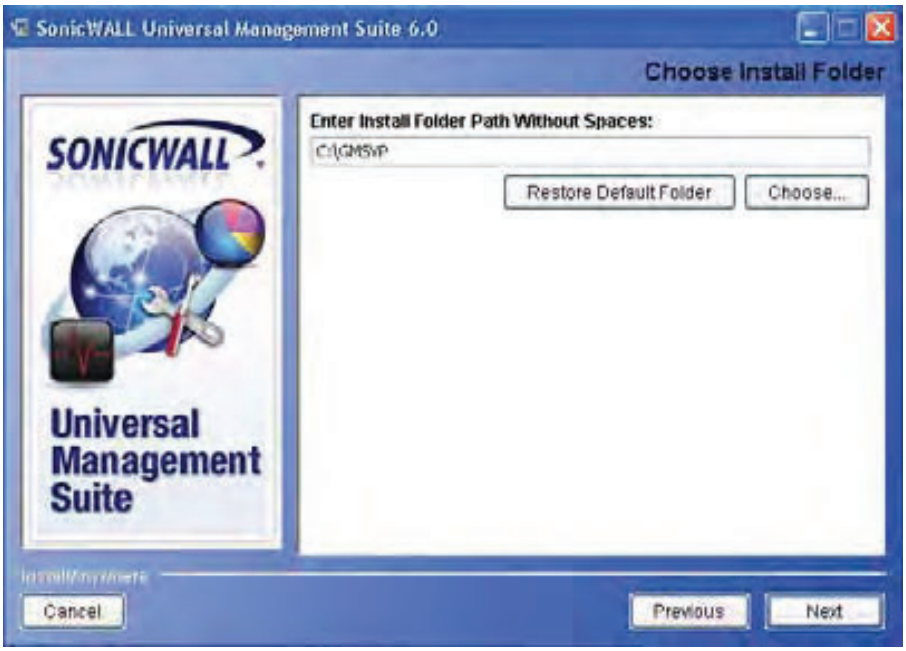

4. In the License Agreement screen, select the radio button next to **I accept the terms of the License Agreement**. Click **Next**.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1588
Keysight, EX-1045
Page 1588

5. Select the path to the folder where you would like to install SonicWALL ViewPoint. You can accept the default path, **C:\GMSVP**, type in a new path, or click the **Choose** button to navigate to the selected folder. When you are finished, click **Next**.

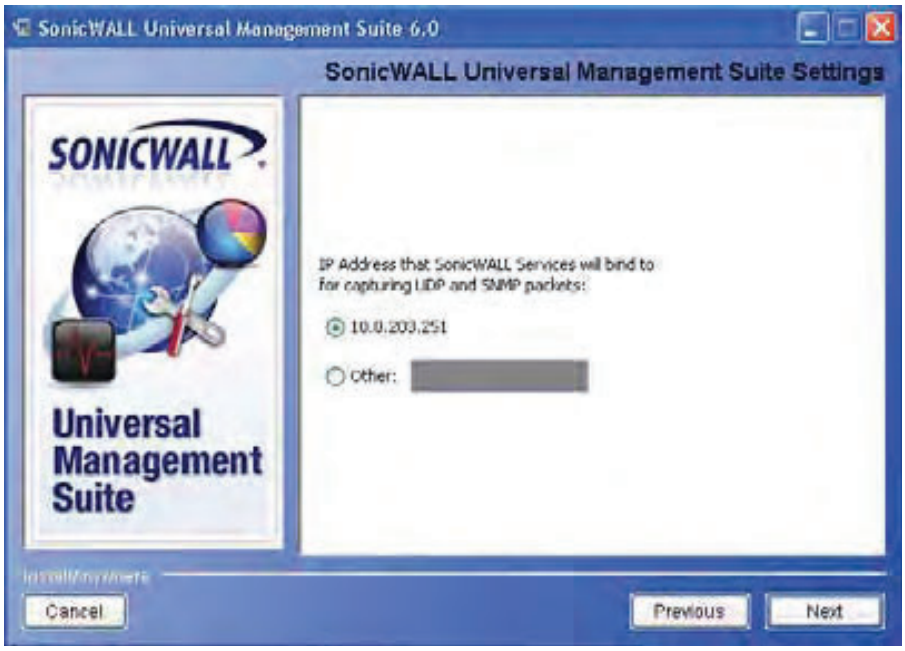



**Tip** Do not include spaces in the SonicWALL ViewPoint installation path.

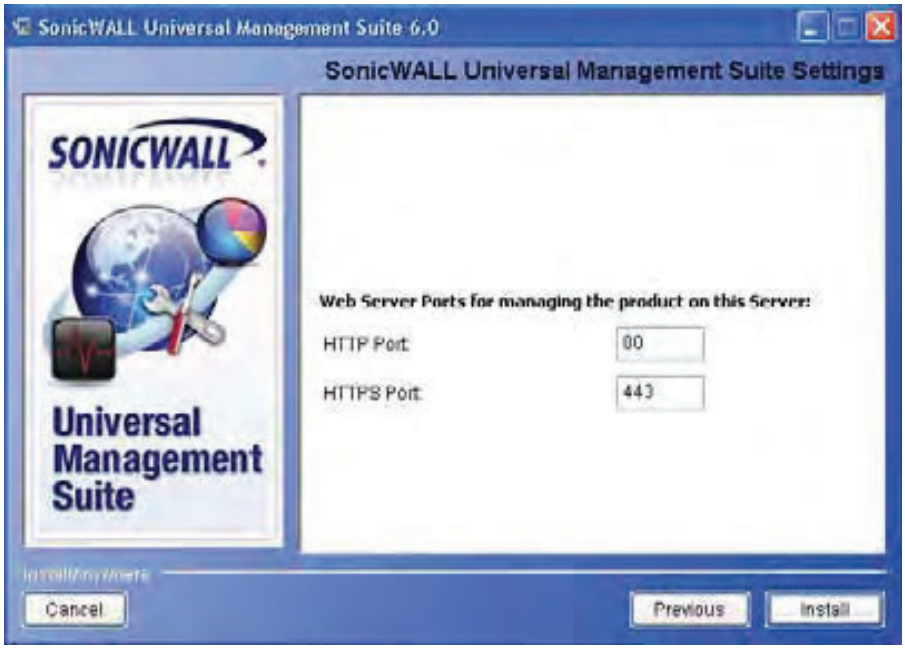


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1589
Keysight, EX-1045
Page 1589

6. Select the IP address you want SonicWALL Services to bind to for capturing syslog and SNMP packets. The default is your management computer IP address. To provide a different IP address, select the radio button next to **Other** and enter the IP address. Click **Next**.




7. In the SonicWALL Universal Management Suite Settings window, enter the Web server ports for HTTP and HTTPS.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1590
Keysight, EX-1045
Page 1590

**Tip** If you receive the message "Cannot bind to the port number specified. Please specify a different one," the port you specified in **Web Server Port** is in use by another program, for example, Internet Information Services (IIS). Specify another unused Web server port, for example, 8080.

**Tip** If you specify a custom port, you will need to modify the URLs you use to access SonicWALL ViewPoint by using the following format: **http://localhost:<port>/sgms/login** (to login from the local host) or **http://<*host_ipaddress*>:<port>/sgms/login** (to login from a remote location). For example, if you specified port 8080, the URL would be **http://localhost:8080/sgms/login** for a local host login, or **http://10.0.93.20:8080/sgms/login** for a remote login.

8. Click **Install**. You may see a Windows Firewall security alert. If you do, click **Unblock**.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1591
Keysight, EX-1045
Page 1591

9. The Installer displays the installation progress during the few minutes required. Upon completion, whether or not the system has Windows Firewall enabled, a dialog is displayed notifying you to either disable the firewall or manually open the syslog and SNMP ports, and to ensure that these ports are open on your network gateway or firewall. Click **OK**.




10. The Important Registration Information screen provides the URL and credentials to use to access the SonicWALL ViewPoint Universal Management Host system interface after restarting your system, as well as information about registration.

    The default URL for accessing the interface from the local system is:
    **http://localhost:80/**

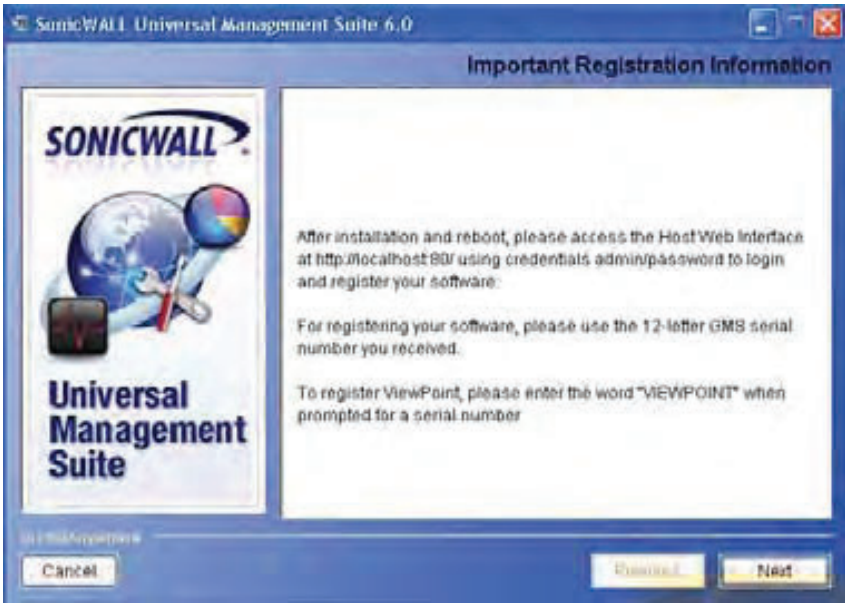
    The default credentials are:
    User name – **admin**
    Password – **password**

    To register for a SonicWALL ViewPoint installation, enter the word **VIEWPOINT** instead of a serial number when you register the product on MySonicWALL.



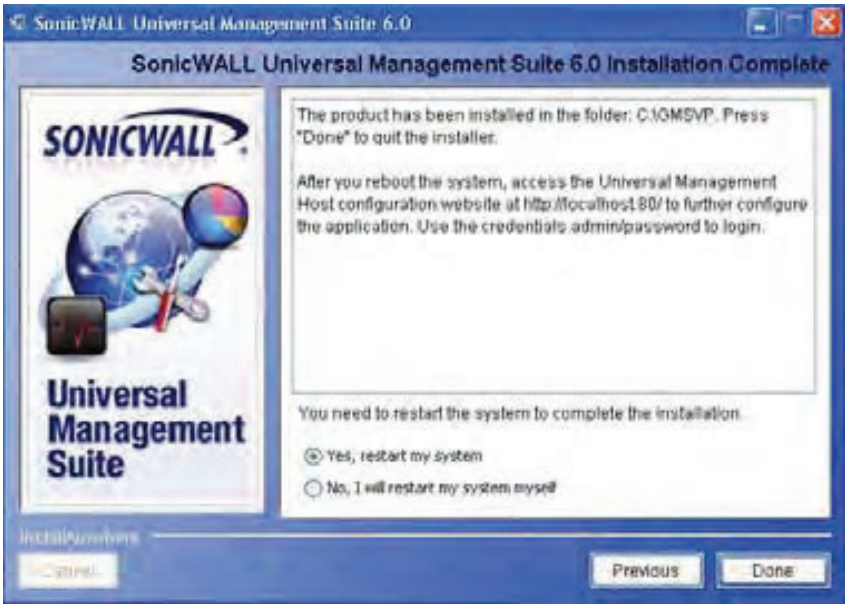
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1592
Keysight, EX-1045
Page 1592

Click **Next**.




11. In the Installation Complete screen, select one of the following options for restarting your system to complete the installation, and then click **Done**:
    - **Yes, restart my system**
    - **No, I will restart my system myself**

**Note** Restarting after installation is required for full functionality.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1593
Keysight, EX-1045
Page 1593

12. After restarting your system, you can access the SonicWALL ViewPoint UMH system interface by either clicking on the new desktop shortcut for **SonicWALL Universal Management Suite 6.0** or by pointing your browser at **http://localhost:80/**.
13. Your default Web browser will launch **http://localhost:80/appliance/login**.
14. Login using the username **admin** and the password **password**.
15. You will be prompted to change your password.

**Note** You are forced to change your password the first time you login.

# Upgrading SonicWALL ViewPoint 5.1 to 6.0

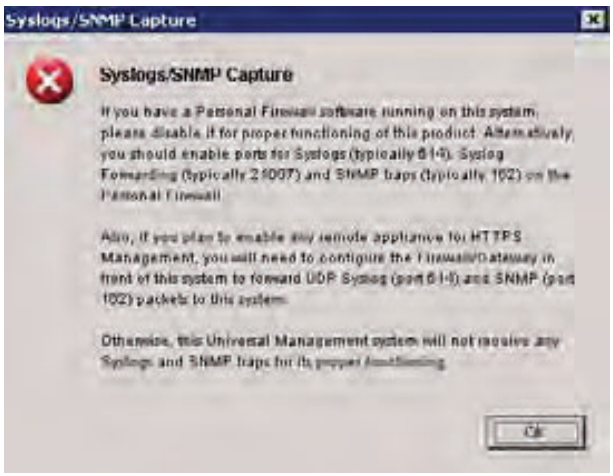
To upgrade from SonicWALL ViewPoint 5.1 to 6.0 using the the Universal Management Suite 6.0 single binary installer, perform the following steps:

1. Log on to your SonicWALL ViewPoint management computer as **administrator** (Windows). Launch the SonicWALL Universal Management Suite 6.0 installer, by double-clicking the file **sw_gmsvp_win_eng_6.0.xxxx.xxxx.exe** (where "xxxx" represent the exact version numbers). It may take several seconds for the InstallAnywhere self-extractor to initialize.
2. In the Introduction screen, click **Next**.
3. In the License Agreement screen, select the radio button next to **I accept the terms of the License Agreement**. Click **Next**.
4. When the installer detects that SonicWALL ViewPoint 5.1 is currently installed on the system, a notification is displayed. Click **Install** to continue the upgrade.
5. The installer begins installing the files, using the existing installation folder, IP address to which SonicWALL Services bind for capturing syslog and SNMP packets, and Web port settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1594
Keysight, EX-1045
Page 1594

6. The Installer displays the installation progress during the few minutes required. Upon completion, whether or not the system has Windows Firewall enabled, a dialog is displayed notifying you to either disable the firewall or manually open the syslog and SNMP ports, and to ensure that these ports are open on your network gateway or firewall. Click **OK**.




7. The Important Registration Information screen provides the URL for access to the SonicWALL ViewPoint Universal Management Host system interface after upgrade completion, as well as information about registration.

   The default URL for accessing the interface from the local system is:
   **http://localhost:80/**

   The default credentials are:
   User name – **admin**
   Password – **password**

   To register for a SonicWALL ViewPoint installation, enter the word **VIEWPOINT** instead of a serial number when you register the product on MySonicWALL.

   Click **Next**.

8. The final installer screen contains the path of the installation folder, and warns you that the Universal Management Suite Web page will be launched next. Click **Done**.

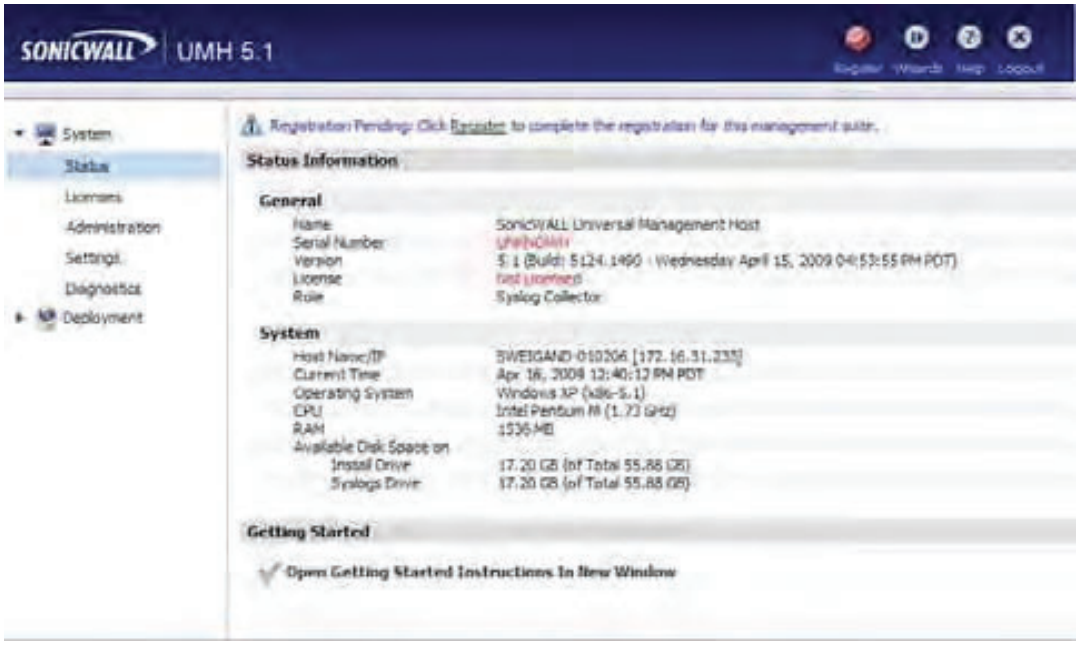
   In the SonicWALL ViewPoint login page, enter the same credentials for **User** and **Password** that you had in your earlier version prior to the upgrade.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1595
Keysight, EX-1045
Page 1595

# Registering SonicWALL ViewPoint

SonicWALL ViewPoint registration is performed using the Universal Management Host system management interface. The first time you log into the system interface, the System > Status page will display a **Registration Pending** notification at the top of the screen, and the **Register** button will be available in the top right corner of the interface.




SonicWALL ViewPoint must be registered before you can use it. To complete registration, SonicWALL ViewPoint must have access to the Internet. The SonicWALL ViewPoint registration process sends your registration information to the MySonicWALL registration site. When registration is completed, SonicWALL ViewPoint will be licensed on your system.

**Note** MySonicWALL registration information is not sold or shared with any other company.

To register SonicWALL ViewPoint, perform the following steps:

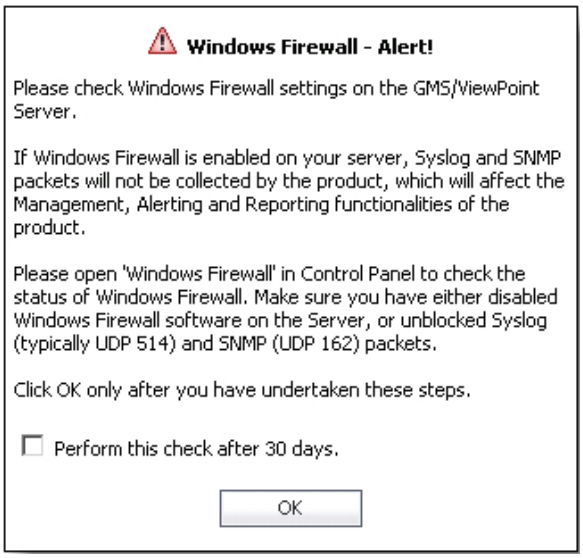
1. In a browser, log in to the system management interface (**http://<*host*>:80/appliance/login)**. If this is the first time you have logged in after running the Installer and rebooting, you will be required to change the password for the admin account. Enter the new password in the appropriate fields and then click **Submit**.



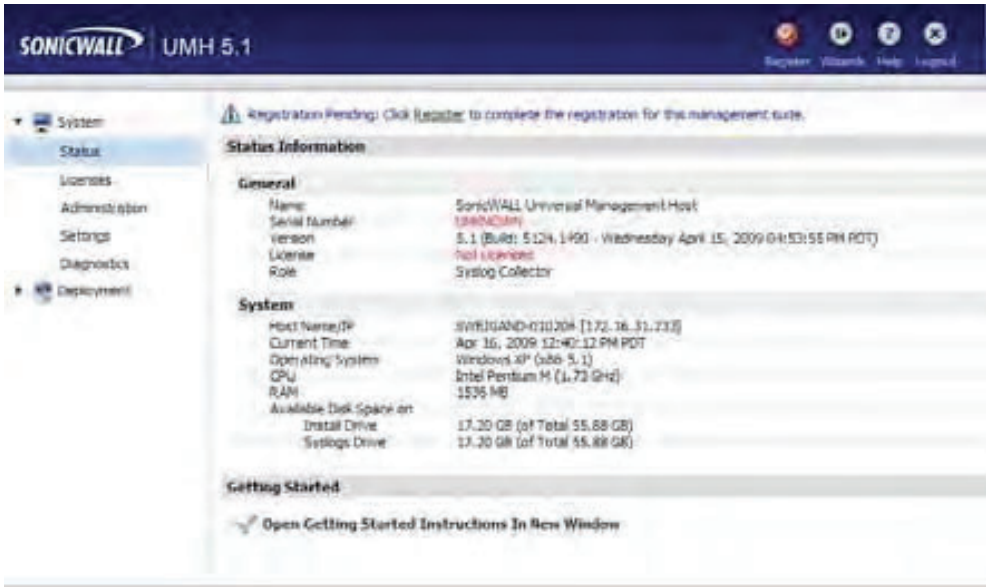
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1596
Keysight, EX-1045
Page 1596

2. If the software detects that the Windows Firewall is enabled on the system, a warning dialog box is displayed on top of the System > Status page. To receive syslog and SNMP packets, either disable the Windows Firewall or configure it to open these ports (default syslog port UDP 514 and default SNMP port UDP 162). When ready, click **OK**.

   Optionally, you can select the **Perform this check after 30 days** checkbox if you do not plan to disable the Windows Firewall immediately, and do not wish to see this warning every time you login. The check for Windows Firewall cannot be disabled completely, and if you leave it running you will see this alert after the 30-day delay. You can repeat the delay as many times as needed.




3. In the System > Status page, click the **Register** button.



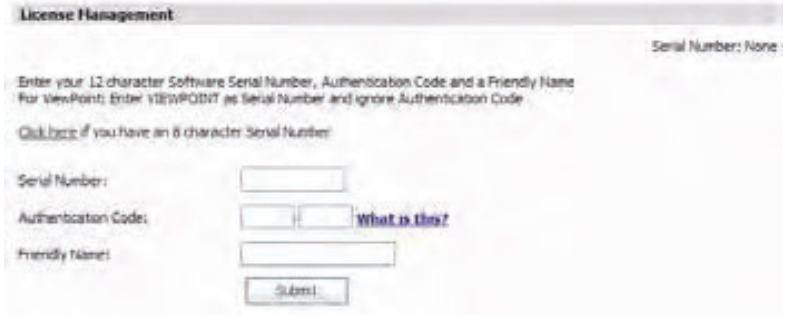



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1597
Keysight, EX-1045
Page 1597

4. In the License Management page, type your MySonicWALL user name and password and then click **Submit**.




5. In the next License Management page, type **VIEWPOINT** (all capital letters) into the **Serial Number** field and leave the **Authentication Code** fields blank. Type a descriptive name for the system into the **Friendly Name** field and then click **Submit**.




**Note** The Friendly Name for this system will also be used as the name for the SonicWALL ViewPoint deployment. As you register SonicWALL appliances on MySonicWALL, you will have the option of adding them to this deployment for SonicWALL ViewPoint reporting.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1598
Keysight, EX-1045
Page 1598

6. In the next License Management page, click **Continue**. This completes the registration process.




When registration is complete, the Deployment > Roles page is displayed. Although there is only one possible role for a SonicWALL ViewPoint deployment, you must still configure certain fields on this page and then click **Update** to fully activate the application. For instructions on configuring these settings, see the "Configuring the Deployment Role" section on page 32.

# Configuring Deployment Settings

This section describes the settings available on the **Deployment > Settings** page of the UMH system management interface, available by default at:
**http://localhost/appliance**

## Configuring Web Port Settings

To change the Web port settings, perform the following steps:

1. On the **Deployment > Settings** page under **Web Port Configuration**, to use a different port for HTTP access to the SonicWALL ViewPoint, type the port number into the **HTTP Port** field. The default port is 80.




2. To use a different port for HTTPS access to the SonicWALL ViewPoint, type the port number into the **HTTPS Port** field. The default port is 443.
3. Click **Update** to apply the Web port settings.

**Note** Changing the Web port settings will cause the system to restart.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1599
Keysight, EX-1045
Page 1599

4. After the appliance restarts, use the new port to access the "appliance" or SonicWALL ViewPoint management interface. For example:
   - If you changed the HTTP port to 8080, use the URL: **http://<*IP Address*>:8080/appliance/**
   - If you changed the HTTPS port to 4430, use the URL: **http://<*IP Address*>:4430/appliance/**

# Configuring SMTP Settings

The SMTP settings are used for sending email alerts to the SonicWALL ViewPoint administrator. To configure the SMTP settings, perform the following steps:

1. On the **Deployment > Settings** page under **SMTP Configuration**, enter the IP address of the SMTP server into the **SMTP server** field.




2. In the **Sender address** field, enter the email address that will appear as the 'From' address when email alerts are sent to the administrator.
3. In the **Administrator address** field, enter a valid email address for the administrator who will receive email alerts.
4. Click **Update** to apply the SMTP settings.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1600
Keysight, EX-1045
Page 1600

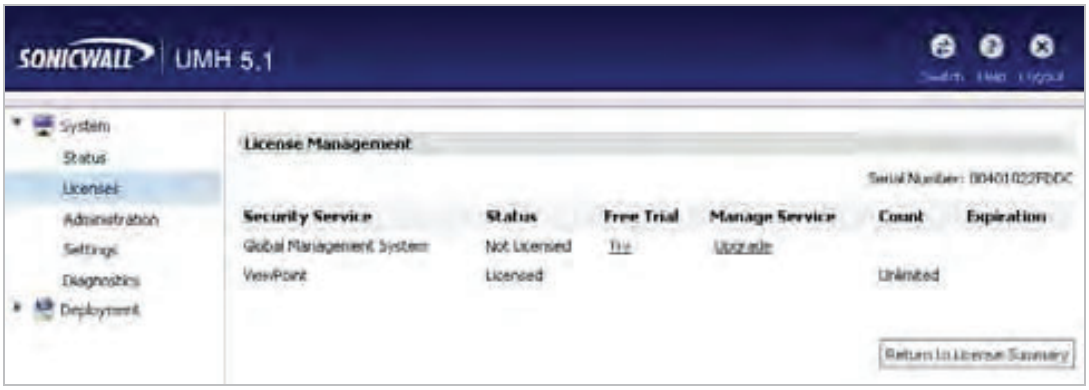
# Upgrading from ViewPoint to GMS

SonicWALL ViewPoint installations have the option of upgrading to SonicWALL GMS without reinstalling. You can start a 30-day Free Trial of SonicWALL GMS by clicking a button or link in either the ViewPoint or Universal Management Host interface and following a simple procedure. When you are ready to finalize the upgrade, your SonicWALL reseller can provide you with the license key for a seamless transition to SonicWALL GMS.

When five or more registered devices are connected to SonicWALL ViewPoint reporting, the **Try GMS Free - 30 Days** button appears next to the tabs at the top of the SonicWALL ViewPoint management interface.




You can also start the Free Trial by clicking **Manage Licenses** on the **System > Licenses** page of the Universal Management Host interface, and then clicking the **Try** link.




For details on enabling the SonicWALL GMS Free Trial and purchasing the SonicWALL GMS upgrade license, see the following sections:

- "Enabling the GMS Free Trial from ViewPoint" section on page 357
- "Enabling the GMS Free Trial from the UMH Interface" section on page 359
- "Completing the Free Trial Upgrade" section on page 360
- "Configuring Appliances for GMS Management" section on page 364
- "Purchasing a SonicWALL GMS Upgrade" section on page 366



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1601
Keysight, EX-1045
Page 1601

# Enabling the GMS Free Trial from ViewPoint

When five or more devices are connected to SonicWALL ViewPoint reporting, the **Try GMS Free - 30 Days** button appears next to the tabs at the top of the SonicWALL ViewPoint management interface.

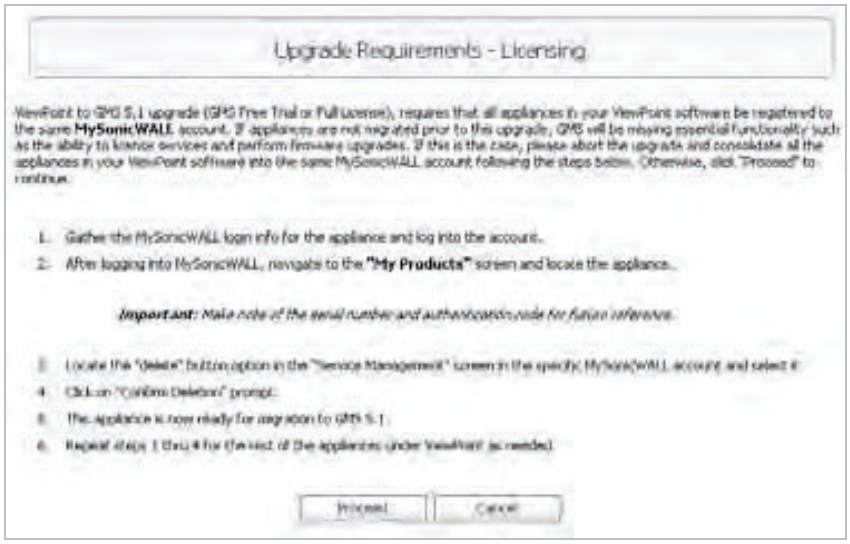
To find out how many devices your SonicWALL ViewPoint installation is handling, log in to MySonicWALL and navigate to the **My Products** page. Click on the link for your SonicWALL ViewPoint installation to get to the **Service Management** page, and scroll to the bottom. You will see the list of appliances under **Associated Products**.

To enable the 30-day SonicWALL GMS Free Trial from the SonicWALL ViewPoint management interface, perform the following steps:

1. In the SonicWALL ViewPoint management interface, click the **Try GMS Free - 30 Days** button next to the tabs at the top of the page.



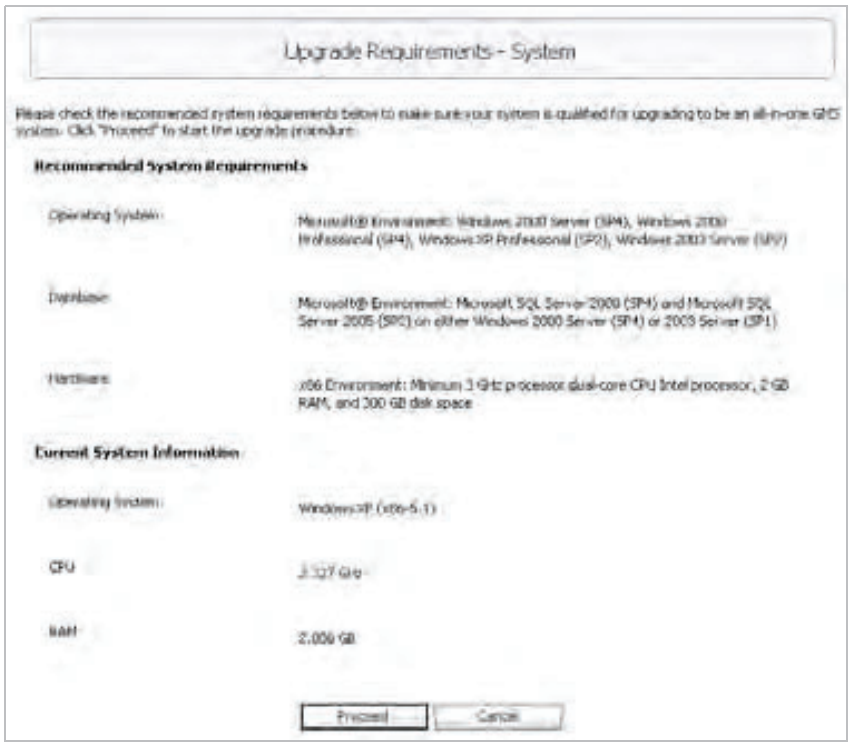

2. The Viewpoint Upgrade Tool launches and guides you through the process of installing the Free Trial or Upgrade. The tool displays the **Upgrade Requirements – Licensing** screen. Before migrating to GMS 5.1, ensure that all appliances under Viewpoint reporting are registered to the same MySonicWALL account. Follow the steps provided in the screen, and then click **Proceed**.



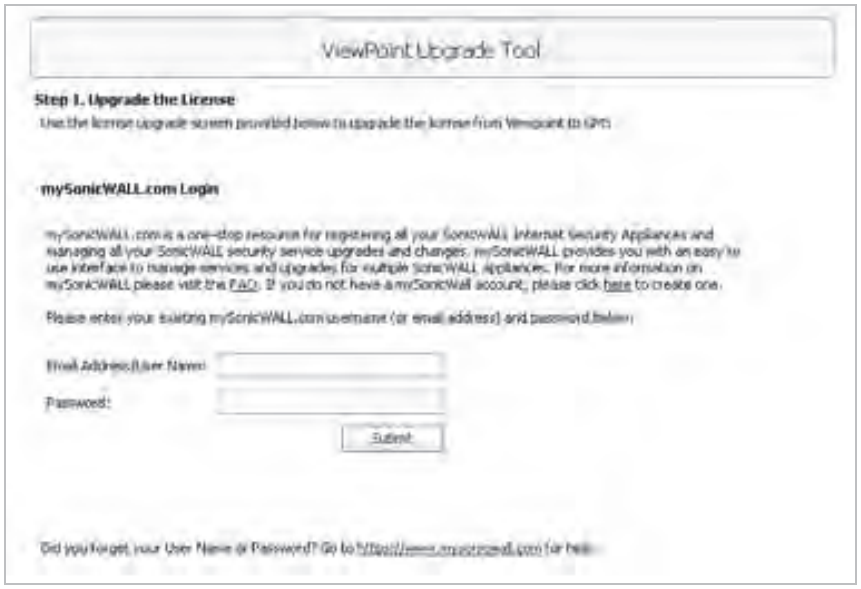



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1602
Keysight, EX-1045
Page 1602

3. The **Upgrade Requirements – System** screen displays the recommended operating system, database, and hardware system requirements. Click **Proceed**.



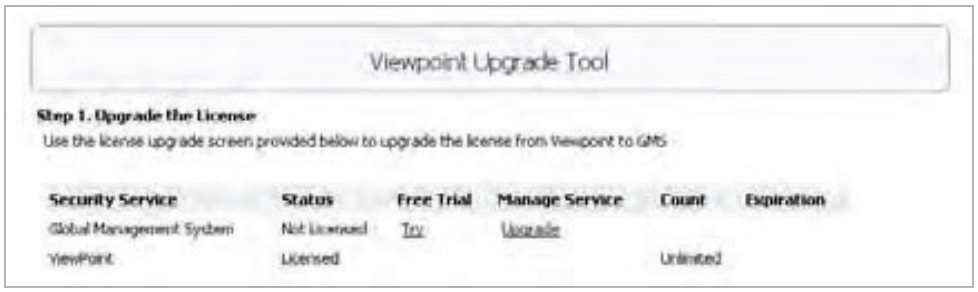

4. The ViewPoint Upgrade Tool displays the login screen for MySonicWALL. Enter your MySonicWALL credentials and click **Submit**.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1603
Keysight, EX-1045
Page 1603

5. In the next ViewPoint Upgrade Tool page, click the **Try** link in the **Free Trial** column for Global Management System.

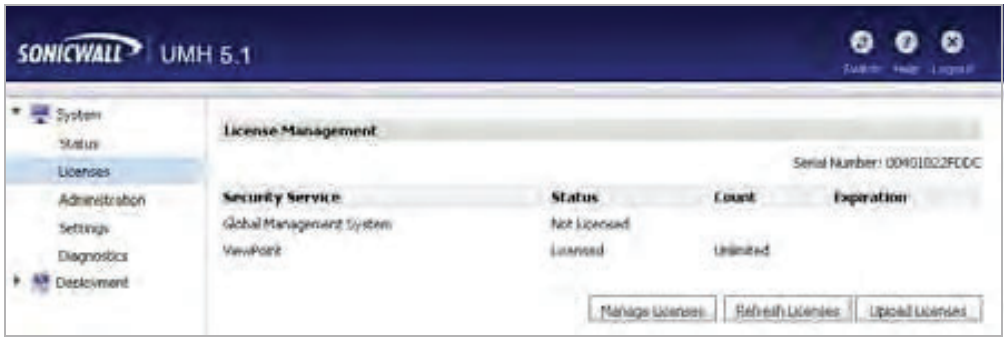



6. From this point, the upgrade process continues with the same steps for access from either the SonicWALL ViewPoint interface or the Universal Management Host interface. To continue the procedure, perform the steps in the "Completing the Free Trial Upgrade" section on page 360.

# Enabling the GMS Free Trial from the UMH Interface

To enable the 30-day Free Trial of SonicWALL GMS from the Universal Management Host interface on your SonicWALL ViewPoint system, perform the following steps:

1. In the Universal Management Host interface, navigate to the System > Licenses page and click **Manage Licenses**.

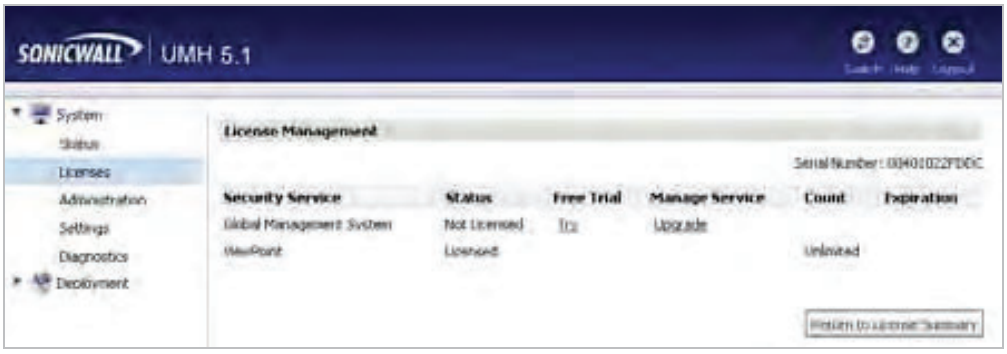



2. If you are not already logged into MySonicWALL, the MySonicWALL login screen is displayed. Enter your MySonicWALL credentials in the appropriate fields and log in.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1604
Keysight, EX-1045
Page 1604

3. On the next page, click the **Try** link in the **Free Trial** column for Global Management System.




4. From this point, the upgrade process continues with the same steps for access from either the SonicWALL ViewPoint interface or the Universal Management Host interface. To continue the procedure, perform the steps in the "Completing the Free Trial Upgrade" section on page 360.

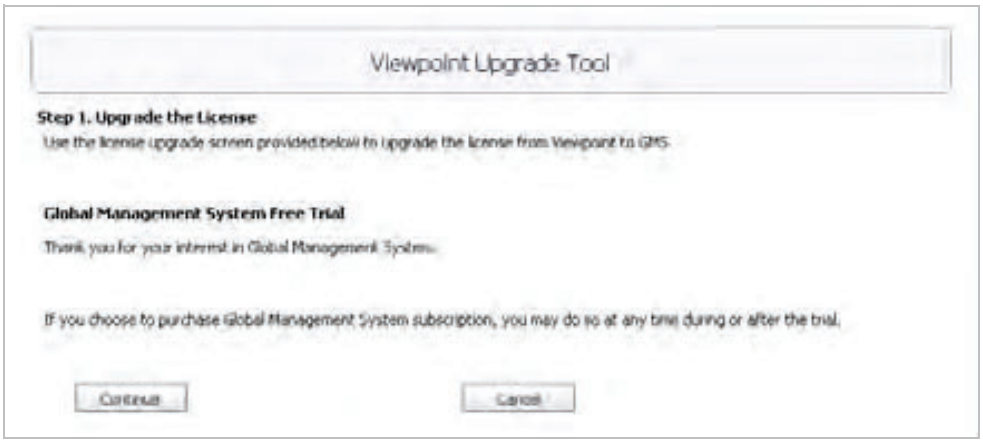
# Completing the Free Trial Upgrade

This procedure provides the common upgrading steps for access from either the SonicWALL ViewPoint interface or the Universal Management Host interface. To get to this point in the process, follow the steps described in one of the two preceding sections:

- "Enabling the GMS Free Trial from ViewPoint" section on page 357
- "Enabling the GMS Free Trial from the UMH Interface" section on page 359

To continue the upgrade, perform the following steps:

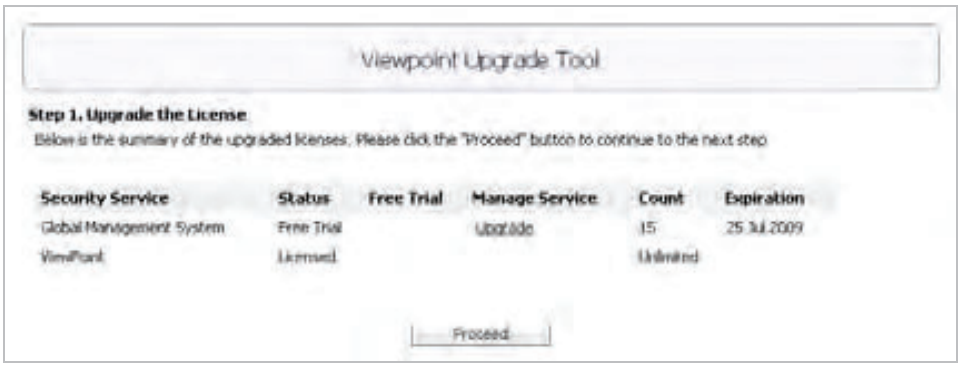
1. In the ViewPoint Upgrade Tool page, click the **Continue** button.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1605
Keysight, EX-1045
Page 1605

2. The next screen provides a summary of GMS and ViewPoint status. Verify that the **Try** link for the Free Trial is gone and only the **Upgrade** link remains. The **Expiration** column displays the expiration date of your Free Trial. You can click the **Upgrade** link at any time during the Free Trial to purchase the SonicWALL GMS upgrade. Click **Proceed**.




3. In the next ViewPoint Upgrade Tool page, you begin the configuration for SonicWALL GMS instep 2 of the upgrade process. This page displays two sections:

   **Automatic Configuration** – Contains a list of SonicWALL UTM or CSM appliances in your ViewPoint installation. These appliances will be automatically configured for SonicWALL GMS management.

   **Manual Configuration** – Contains a list of SonicWALL Aventail, SSL-VPN, or CDP appliances in your ViewPoint installation. You must manually configure these appliances for SonicWALL GMS management. See the "Configuring Appliances for GMS Management" section on page 364 for detailed instructions on enabling SonicWALL GMS management on these appliances.

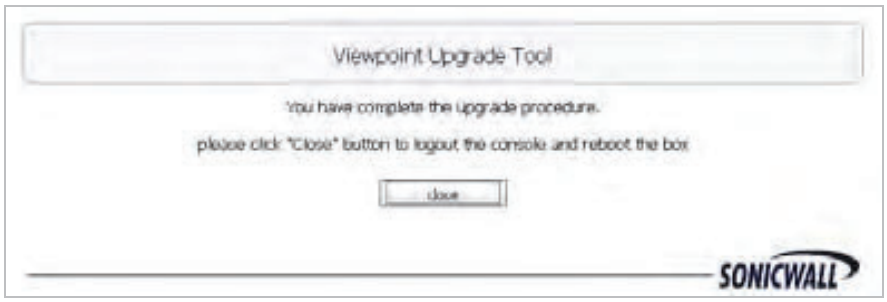


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1606
Keysight, EX-1045
Page 1606

When ready, click **Proceed**.




4. When the configuration finishes, the ViewPoint Upgrade Tool displays the completion dialog box. Click **Close** to log out of the console and restart the system.





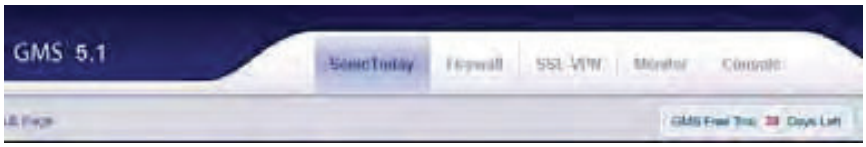

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1607
Keysight, EX-1045
Page 1607

5. The GMS login page appears and requests that you reboot the system. Reboot the system. If a reboot is not performed, you may encounter problems with the correct IP Address appearing.

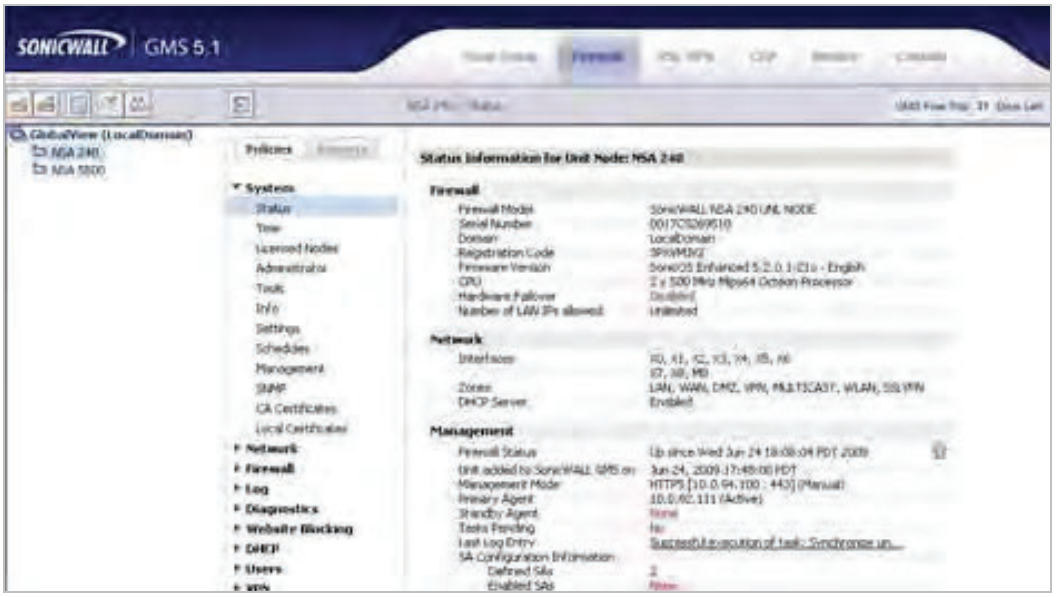



6. After rebooting, log in with your ViewPoint credentials.

   When you log in, you will see a button displaying the number of days left in your Free Trial at the top of the page.




7. On the System > Status page for connected appliances, you can view the log entries for task synchronization and automatic addressing mode, related to the GMS configuration.






CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1608
Keysight, EX-1045
Page 1608

# Configuring Appliances for GMS Management

To manually configure the appliances listed in the Manual Configuration section of the ViewPoint Upgrade Tool page (see Step 3. on page 361), perform the following steps for each appliance:

1. In the SonicWALL GMS management interface, click the tab at the top of the page that corresponds to the type of appliance, such as **SSL-VPN** or **CDP**.
2. In the left pane, right-click one of the listed appliances and select **Modify Unit**.
3. In the Modify Unit screen in the right pane, copy the appliance IP address in the **Managed Address** section to your clipboard, or make a note of it.

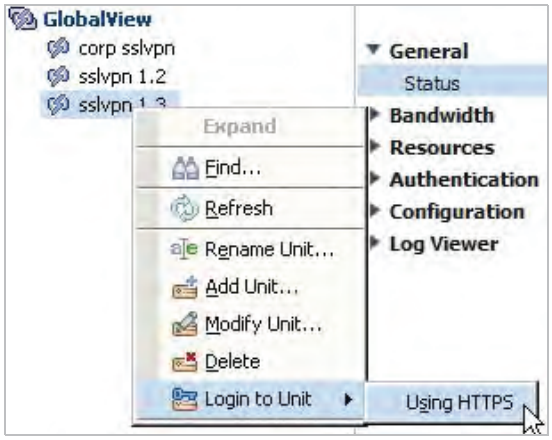



4. Click **Cancel**.

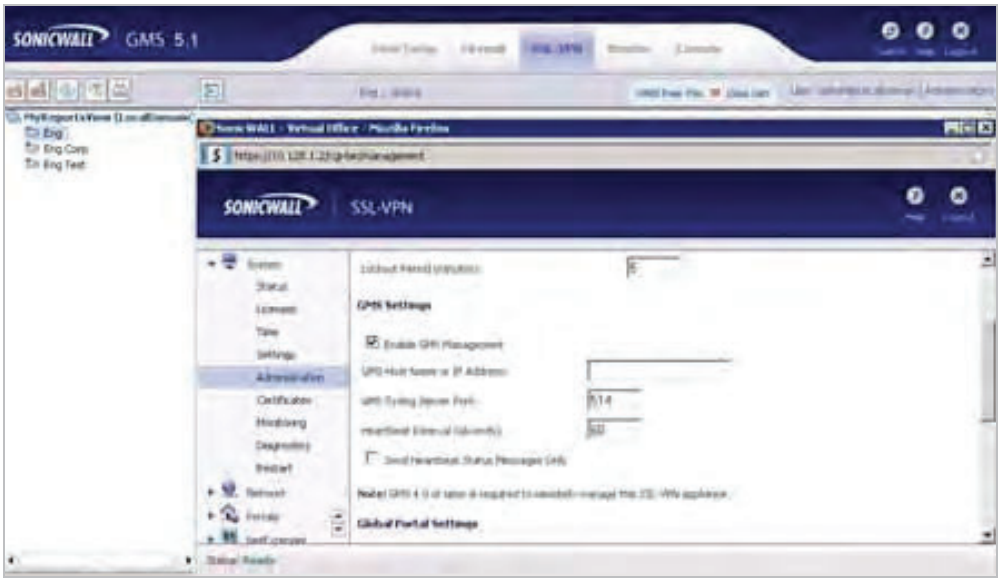


CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1609
Keysight, EX-1045
Page 1609

5. In the left pane, right-click the same appliance and select **Login to Unit > Using HTTPS**.




6. In the appliance management interface, navigate to the System > Administration page.




7. Under **GMS Settings**, select the **Enable GMS Management** checkbox, or verify that it is selected.
8. In the **GMS Host Name or IP Address** field, paste or type the appliance IP address that you obtained from the Modify Unit screen in Step 3.
9. Click the **Accept** button at the top of the appliance interface screen.
10. Click the **Logout** button in the top right corner of the appliance interface screen.
11. Repeat these steps for each appliance listed in the Manual Configuration section of the ViewPoint Upgrade Tool page.



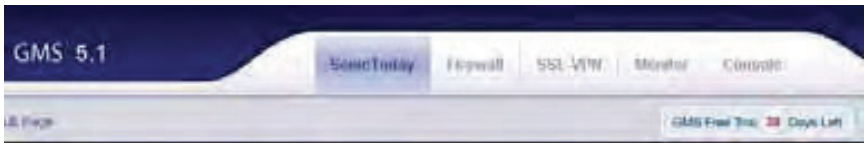
CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1610
Keysight, EX-1045
Page 1610

# Purchasing a SonicWALL GMS Upgrade

You can purchase an upgrade to SonicWALL GMS at any time during the 30-day Free Trial.

To purchase the SonicWALL GMS license, perform the following steps:

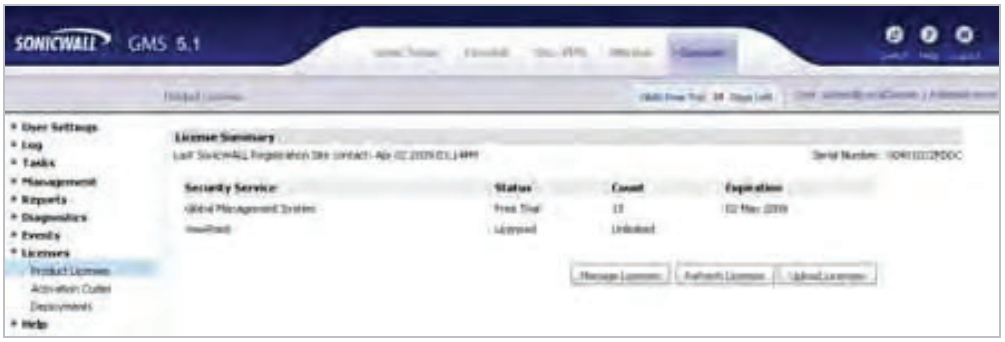
1. In the SonicWALL GMS interface, click the **GMS Free Trial X Days Left** button, where X is the number of days left in the Free Trial.




2. In the **Buy GMS** page, click **I want to upgrade to GMS now**.



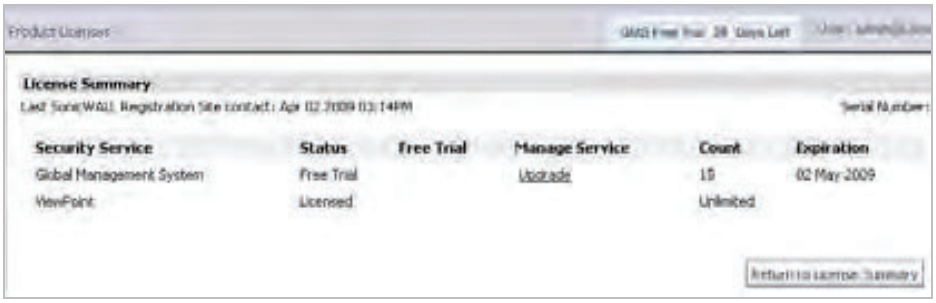

3. The Console > Licenses > Product Licenses page is displayed. Click **Manage Licenses**.



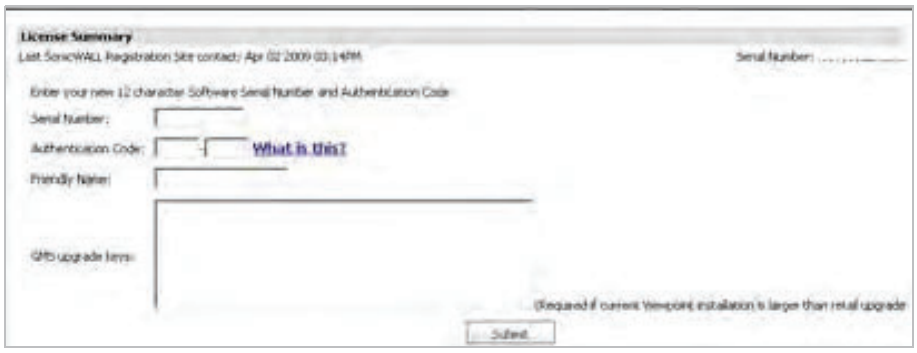



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1611
Keysight, EX-1045
Page 1611

4. In the next page, in the **Manage Service** column for Global Management System, click the **Upgrade** link.




5. The next page has **Serial Number** and **Authentication Code** fields for SonicWALL GMS. You must contact your SonicWALL reseller to complete the purchase and obtain the 12-character serial number and authentication code. Type in the values to the **Serial Number** and **Authentication Code** fields.




6. Enter a descriptive name for the SonicWALL GMS installation into the **Friendly Name** field. This name will appear in your MySonicWALL account.

7. If your SonicWALL ViewPoint installation currently handles more than 10 appliances, when you upgrade to SonicWALL GMS you will need to purchase additional SonicWALL GMS license(s) to manage the extra appliances. The standard "10-node" SonicWALL GMS license provided with the Free Trial supports up to 10 managed appliances. Enter the license keys for any additional SonicWALL GMS licenses into the **GMS upgrade keys** text box, one key per line.

8. Click **Submit**. The License page is displayed, showing that SonicWALL GMS is now licensed.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1612
Keysight, EX-1045
Page 1612

# Miscellaneous Procedures and Troubleshooting Tips

This section contains miscellaneous SonicWALL ViewPoint procedures and troubleshooting tips.

## Miscellaneous Procedures

This section contains information on procedures that you may need to perform. Select from the following:

- It is highly recommended that you regularly back up the SonicWALL ViewPoint data. For more information, see "Backing up SonicWALL ViewPoint Data" on page 368.
- SonicWALL ViewPoint requires Mixed Mode authentication when using SQL Server 2000. To change the authentication mode, see "Changing the SQL Server Authentication Mode" on page 369.
- If you are reinstalling SonicWALL ViewPoint, preserving the previous configuration settings can save a lot of time. To reinstall SonicWALL ViewPoint using an existing SonicWALL ViewPoint database, see "Reinstalling SonicWALL ViewPoint Using an Existing Database" on page 369.
- If you need to uninstall SonicWALL ViewPoint from a server, it is important to do it correctly. To uninstall SonicWALL ViewPoint, see "Uninstalling SonicWALL Universal Management Suite and Its Database" on page 369.

### Backing up SonicWALL ViewPoint Data

SonicWALL ViewPoint stores its configuration data in the SGMSDB database. It is important to back up this database and the individual SonicWALL ViewPoint databases (sgmsvp_yyyy_mm_dd) on a regular basis.

The Console > Management > Database Maintenance page provides the necessary support for backing up and restoring the MySQL database that is bundled with SonicWALL UMS. For more information, see the "Database Maintenance" section on page 66.

If you are using SQL Server, this can be accomplished by backing up the entire SQL Server using the database backup tool. When using this tool, there is no need to stop the SonicWALL ViewPoint services for database backup. However, make sure that the backup occurs when SonicWALL ViewPoint activity is the lowest and that the backup operation schedule does not clash with the SonicWALL ViewPoint scheduler.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1613
Keysight, EX-1045
Page 1613



**Note** It is also recommended to regularly back up the entire contents of the SonicWALL ViewPoint directory, the sgmsConfig.xml file.

## Changing the SQL Server Authentication Mode

SonicWALL ViewPoint requires the Mixed Mode authentication mode. To change the authentication mode from Windows Mode to Mixed Mode, follow these steps:

1. Start the Microsoft SQL Server Enterprise Manager.
2. Right-click the appropriate SQL Server Group and select **Properties** from the pop-up menu.
3. Click the **Security** tab.
4. Change the Authentication mode from **Windows only** to **SQL Server and Windows**.
5. Click **OK**.

## Reinstalling SonicWALL ViewPoint Using an Existing Database

If you need to reinstall SonicWALL ViewPoint, but want to preserve the settings in an existing SonicWALL ViewPoint database, follow these steps:

1. Install a new database, using the same username and password that you used for the existing SonicWALL ViewPoint database.
2. Install SonicWALL ViewPoint using this new database.
3. Stop all SonicWALL ViewPoint services.
4. Open the sgmsConfig.xml and web.xml files with a text editor. Change the values for the dbhost and dburl parameters to match the existing SonicWALL ViewPoint database.
5. Restart the SonicWALL ViewPoint services.
6. Uninstall the new database.

## Uninstalling SonicWALL Universal Management Suite and Its Database

This section describes how to uninstall SonicWALL Universal Management Suite and its components. Select from the following:

- To uninstall SonicWALL Universal Management Suite on the Windows platform, see "Windows" on page 370.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1614
Keysight, EX-1045
Page 1614

- To uninstall SonicWALL Universal Management Suite databases from Microsoft SQL Server 2000, see "MS SQL Server 2000" on page 370.

### Windows

To uninstall SonicWALL Universal Management Suite from a Windows system, follow these steps:

1. Click **Start**, point to **Settings**, and click **Control Panel**.
2. Double-click **Add/Remove Programs**. The Add/Remove Programs Properties window displays.
3. Select **SonicWALL Universal Management Suite** and click **Change/Remove**. The SonicWALL Universal Management Suite Uninstall program starts.
4. Follow the on-screen prompts.
5. Restart the system. SonicWALL Universal Management Suite is uninstalled.

### MS SQL Server 2000

To uninstall or remove the SonicWALL Universal Management Suite databases in the MS SQL Server 2000, you can execute the following DOS command from any SonicWALL Universal Management Suite server:

```
osql -U username -P password -S dbHost_IP -q "drop database SGMSDB"
osql -U username -P password -S dbHost_IP -q "drop database sgmsvp_yyyy_mm_dd"
```

Or you can use the MS SQL Server's Enterprise Manager and delete the SGMSDB and sgmsvp_ databases.

# Troubleshooting Tips

This section contains SonicWALL ViewPoint troubleshooting tips.

## Changing the Default Syslog Server Port Number

By default, the SonicWALL ViewPoint syslog server default port number is 514 on Windows systems. To change the port number, follow these steps:

1. Open the sgmsConfig.xml file with a text editor.
2. Add the following line to the end of the file before the </Configuration> section:

Parameter name="syslog.syslogServerPort" value="*port_number*"

where *port_number* is the new port number.

3. Save the file and exit.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1615
Keysight, EX-1045
Page 1615

# Installing the Java Plug In

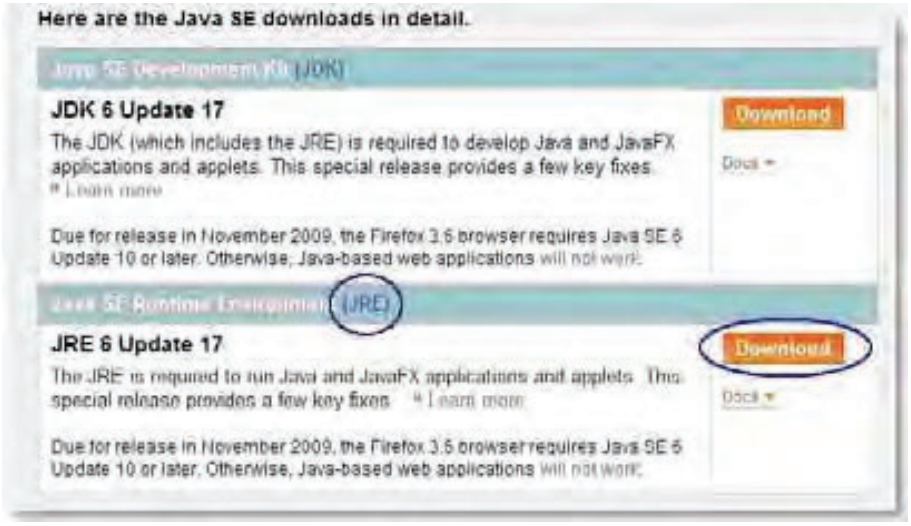
You need Java Plug-in 1.6 or later to access the SonicWALL ViewPoint management interface.

**Tip** The Java Plug-in is automatically installed during the SonicWALL ViewPoint installation. However, you can manually install the Java Plug-in by following these steps.

To manually install the Java Plug-in, perform the following steps:

1. Download the installer from the Internet at:

   http://java.sun.com/javase/downloads/index.jsp




2. Execute the installer.
3. Select the radio button next to **Accept the Terms of the License Agreement**. Click **Next**.
4. Select the radio button next to **Typical** installation and click **Next**.
5. It may take several minutes for the Java Plug-in to install.
6. In the Installation Complete window, click **Finish**.
   - Restart your computer to complete the installation process.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1616
Keysight, EX-1045
Page 1616

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1617
Keysight, EX-1045
Page 1617

# Appendix B

# Technical Tips

This chapter includes the following sections:


- "Log Viewer" section on page 373
- "Real-time Syslog Viewer" section on page 375
- "Forwarding Syslog Data to Another Syslog Server" section on page 376
- "Posting ViewPoint Reporting to Another Web Server for End-User Access" section on page 377


## Log Viewer

The Log Viewer contains detailed information on each transaction that occurred on the SonicWALL appliance. This information is stored for the time that you specified in the configuration settings.



**Note** The Log Viewer displays raw log information for every connection. Depending on the amount of traffic, this can quickly consume a large amount of space in the database. It is highly recommended to be careful when choosing the number of days of information that will be stored. For information about setting the number of days data is stored, see "Enabling Report Table Sorting" on page 72.

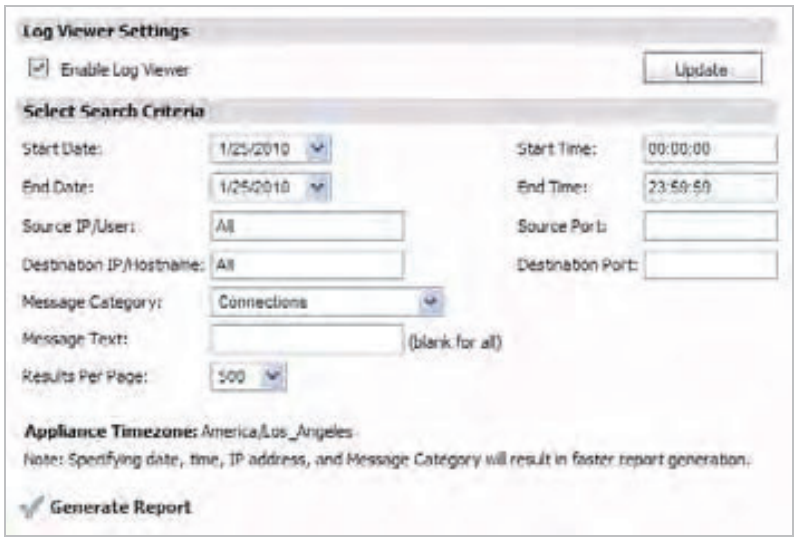
To configure Log Viewer settings for generating a report, perform the following steps:

1. Start and log into SonicWALL ViewPoint.
2. Click the **UTM** or **SSL-VPN** tab.




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1618
Keysight, EX-1045
Page 1618

3. Select a SonicWALL appliance.
4. Expand the **Log Viewer** tree and click **Search**. The Search page displays. Log Viewer must be enabled for the appliance in order to display all the fields on the page.




5. Select **Enable Log Viewer** and then click **Update** to turn on collection of raw data in the database and enable viewing of that log data. This can consume a large amount of space in your database. Review your database space constraints before enabling the log viewer. The maximum number of appliances for which Log Viewer can be enabled is controlled on the Console > Reports > Settings page. See "Controlling the Number of Appliances with Log Viewer Enabled" on page 72.

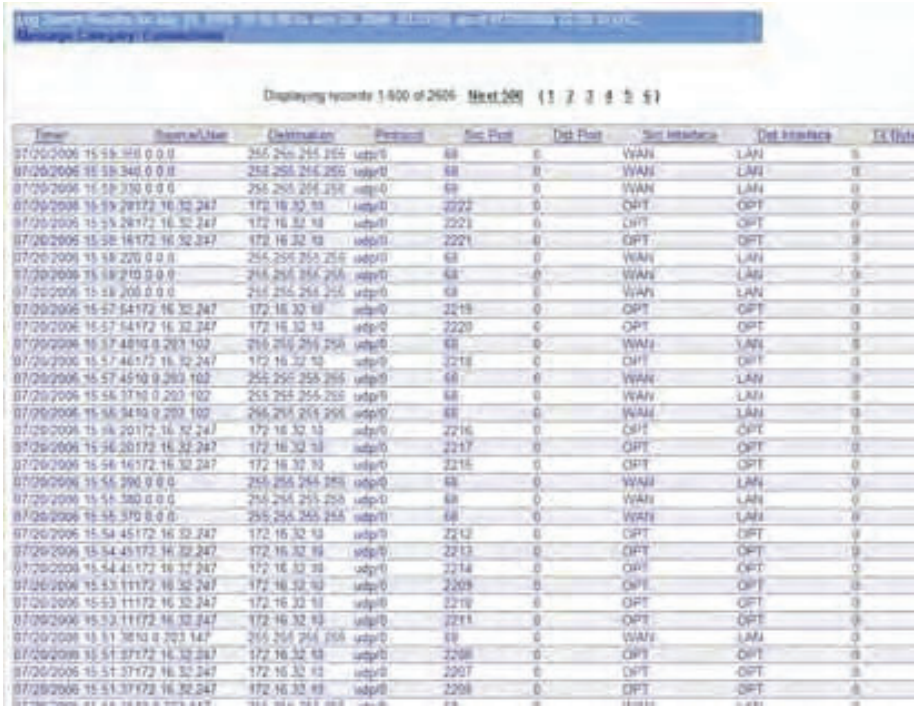
**Note** Custom Reports are available on appliances with Log Viewer enabled. See "Using Custom Reports on UTM Appliances" on page 163.

6. Select the starting date to view from the **Start Date** list box.
7. Enter the starting time of events to view in the **Start Time** field.
8. Select the ending date of events to view in the **End Date** list box
9. Enter the ending time of events to view in the **End Time** field.
10. Enter the source IP address to view in the **Source IP Address** field. To view all IP addresses, enter All.
11. Optionally enter the source port to view in the **Source Port** field.
12. Enter the destination IP address to view in the **Destination IP Address** field. To view all IP addresses, enter All.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1619
Keysight, EX-1045
Page 1619

13. Optionally enter the destination port to view in the **Destination Port** field.
14. Select the type of events to view from the **Message Category** list box.
15. To search for specific message text, type the text into the **Message Text** field.
16. Select the number of entries to display per page from the **Results Per Page** field.
17. Click **Generate Report**. The Log Viewer Results page displays.



# Real-time Syslog Viewer

The real-time syslog utility enables you to diagnose the system by viewing the syslog messages in real time.

**Note** Only use this utility when needed for diagnostic purposes.

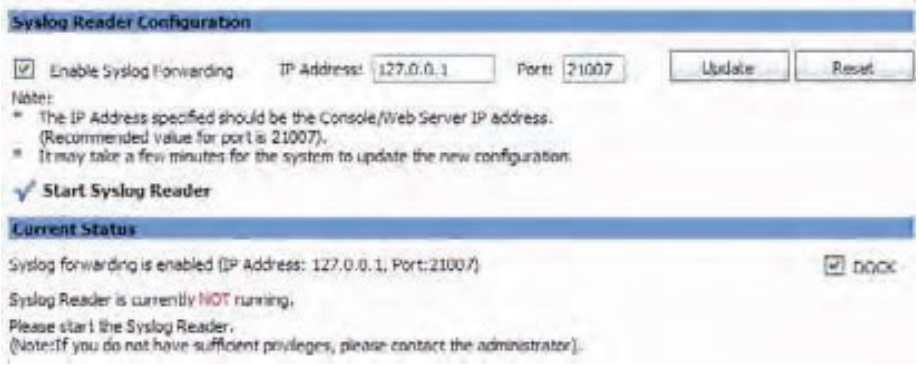
To open the real-time syslog utility, perform the following steps:

1. Start and log into SonicWALL ViewPoint.
2. Click the **UTM** or **SSL-VPN** tab.
3. Expand **Real-Time Viewer** and click **Syslog**. The Real-Time Syslog page appears.
4. If syslog forwarding is not enabled, select **Enable Syslog Forwarding**, set the IP address and port used by the syslog reader, and then click **Update**.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1620
Keysight, EX-1045
Page 1620

5. If the Syslog Reader is not already running, click **Start Syslog Reader**.
6. Click **Start Button** at the bottom of the screen. The Syslog Viewer begins showing the latest syslog entries.




7. To change how many messages are displayed, select a number from the **Number of Messages** list box at the bottom of the screen.




8. To change how often the Syslog Viewer is refreshed, select the time from the **Refresh Time** list box at the bottom of the screen.
9. To stop the viewer, click the **Stop** button.
10. To search for text, use the browser's Find utility.
11. When you are finished, close the Syslog Viewer.

# Forwarding Syslog Data to Another Syslog Server

To forward SonicWALL ViewPoint syslog data to another syslog server, perform the following steps:

1. Open the sgmsConfig.xml file with a text editor.
2. Locate the following line:

   Parameter name ="syslog.forwardToHost" value=""
3. Add the IP address or hostname of the destination syslog server to the value attribute.
4. Save the sgmsConfig.xml file and exit.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1621
Keysight, EX-1045
Page 1621

5. Ensure that at least firmware 6.3.1.0 is running on the SonicWALL appliances.



**Note** To configure SonicWALL ViewPoint to not store the syslog data after it has been forwarded, you must disable the ViewPoint Reporting Module. To do this, open the ViewPoint **Settings** page in the Console Panel, deselect the **Enable Reporting** check box, and click **Update**.

# Posting ViewPoint Reporting to Another Web Server for End-User Access

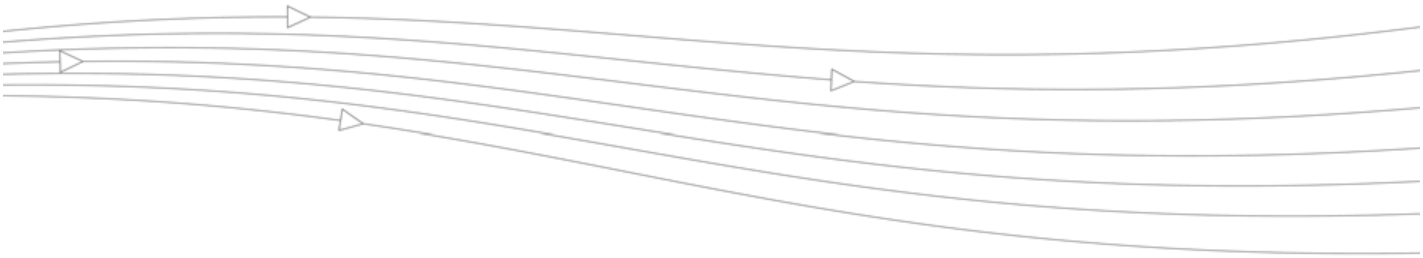
To allow end user access to another web server for end-user access, install the SonicWALL ViewPoint Console in redundant mode.

You can then allow end user access to the redundant Console for viewing ViewPoint Reporting real-time and historical reports. End user access will be isolated from the main Console that is used for managing and configuring SonicWALL appliances.



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1622
Keysight, EX-1045
Page 1622

CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1623
Keysight, EX-1045
Page 1623



# Index


## A

activating
    ViewPoint 341
alert types 100
alerting
    using GEM 97
anti-spyware
    reports 266
applets
    signed 21
Application Firewall reports 281
archive
    in Console>Reports 132
    MDTA 87
    on Console>Reports 81
    report settings 81
    scheduled report 135
    summarizer data 87
Attacks reports 250
authentication code 353
Authentication reports 287

## B

Bandwidth reports 180
benefits
    of compliance reports 144
    of report data management 87
    of SSL VPN reporting 294
browser
    requirements 338

## C

Compliance reports
    configuration 152
    overview 144
compliance reports 144
console
    management settings 61
cover page
    customizing 147
Custom Reports
    Resource Activity 307
customizing
    detailed report 149
    report cover page 147
    summary report 148

## D

dashboard 159
    Dashboard Summary report 159
data
    management 87
database
    backing up 368
    reinstalling with existing db 369
    requirements 337





CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1624
Keysight, EX-1045
Page 1624

deployment
settings 354
detailed report
customizing 149
digital
signature in applet 21
disabling
GEM thresholds 103
domain names
in reports 130

## E

email
report settings 81
enabling
GEM thresholds 103
report table sorting 71
events
GEM overview 97
schedules 98
severities 97
thresholds 98

## F

File Transfer Protocol
See FTP
free trial
for GMS 356
for GMS, configuring appliances 364
for GMS, enabling from UMH interface 359
for GMS, enabling from ViewPoint 357
friendly name 353
FTP reports 225

## G

GEM
alert types 100
configuring 101
default settings 99
overview 97, 98
schedules 98
severites 97
thresholds 98
global view 117
global views 120
GMS
upgrading to from ViewPoint 356
Granular Event Management
See GEM

## H

hardware
requirements 339
help
viewing online help 24
host name resolution 79

## I

inheritance
report settings 138



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1625
Keysight, EX-1045
Page 1625

installing
appendix 335
appliance, firmware support 339
browser requirements 338
database requirements 337
deployment settings 354
hardware requirements 339
Java requirements 338
network requirements 339
on Windows 342
overview 10
system requirements 337
task list 336
Universal Management Suite 342
interfaces
accessing both 11
overview of ViewPoint application 13
switching between 12, 24
TreeControl 20
UMH deployment options 31
UMH deployment role 32
UMH deployment services 36
UMH HTTP(S) settings 34
UMH SMTP settings 34
UMH system administration 28
UMH system diagnostics 30
UMH system interface overview 24
UMH system licenses 26
UMH system settings 25
UMH system software 29
UMH system status 26
IPS
reports 273

## J

Java
plugin 371
requirements 338
java.policy file 21

## L

licensing
requirements 10
SonicWALL appliances for use with ViewPoint 10
ViewPoint on appliances 342
log
viewing 58
Log Viewer 290, 332
log viewer
for SSL VPN appliances 332
logging out 25

## M

Mail Usage reports 231
MDTA 87
MySonicWALL
creating an account 340

## N

name resolution crawler 79
network
requirements 339

## P

pagination
settings 51
password
settings 51
PDF
exporting reports to 144
reports 152
port
syslog 370
profiles
existing 150
scheduled report 149



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1626
Keysight, EX-1045
Page 1626

## R

registering
- procedure 351
- requirements 10
- SonicWALL appliances 341

reporting
- overview 115
- search bar 116, 118

reports
- adding scheduled 135
- Browse Time 128
- By Site to By User navigation 129
- compliance 144
- cover pages 147
- domains or IP addresses 130
- inheritance 138
- no data 156
- PDF format 144
- ROI 128
- searching by dates of 155
- settings 154
- SSL VPN 295
- SSL VPN authentication 330
- SSL VPN bandwidth 301
- SSL VPN custom resource activity 307
- SSL VPN overview 293
- SSL VPN resources 325
- SSL VPN scheduled reports 296
- SSL VPN summarization 297
- viewing Dashboard 159
- views 17
- Web usage, exclusions 53

requirements
- browser 338
- database 337
- hardware 339
- Java 338
- network 339
- system 337

reverse inheritance
- for reporting 139

role
- configuring 32

## S

scheduled reports
- disabling/enabling 128

scheduler
- link to from report page 118

schedules 98
- resending 86

search bar 116, 118
- components of 123
- operators 125

Security Dashboard reports 159
serial number 353
- GMS 114

services
- enabling/disabling 36

Services reports 189
sessions
- managing 65

settings
- inactivity timeout 51
- pagination 51
- password 51
- website exclusion list 53

severities 97
signed applets 21
SMTP
- settings 34, 355




CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1627
Keysight, EX-1045
Page 1627

SNMP
port 339
SQL
authentication mode 369
status
viewing 17
summarizer
configuring reports to create 137
instant summary reports 76
using Summarize Now 76
summary report
customizing 148
syslog
port 339
port number 370
system
requirements 337
system interface
deployment options 31
deployment role 32
deployment services 36
HTTP(S) settings 34
overview 24
SMTP settings 34
system administration 28
system diagnostics 30
system licenses 26
system settings 25
system software 29
system status 26

## T

thresholds 98
timeout
inactivity settings 51
TreeControl
collapsing/hiding the pane 20, 128
menu 20
troubleshooting 368, 370
reports, no data 156

## U

UMH interface
deployment options 31
deployment role 32
deployment services 36
HTTP(S) settings 34
overview 24
SMTP settings 34
system administration 28
system diagnostics 30
system licenses 26
system settings 25
system software 29
system status 26
uninstalling 369
unit view 121
Universal Management Suite
installing 342
upgrading
purchasing GMS upgrade 366
upgrade key 336
ViewPoint to GMS 356
users
managing sessions 65
settings 51

## V

views
global 120
unit 121
Virus Attacks Reports 260
VPN Usage reports 238

## W

Web
port configuration 354
port settings 34



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1628
Keysight, EX-1045
Page 1628

web event consolidation 141
Web Filter reports 209
Web usage
    exclusion list 53
Web Usage reports 191
Windows Firewall
    disabling 339, 352



CS - 1038
Cisco Systems, Inc. v. Centripetal Networks, Inc.
Page 1629
Keysight, EX-1045
Page 1629

**SonicWALL, Inc.**
2001 Logic Drive
San Jose CA 95124-3452
T +1 408.745.9600
F +1 408.745.9300
www.sonicwall.com

**PN: 232-001802-00 Rev A 1/2010**

©2010 SonicWALL, Inc. is a registered trademark of SonicWALL, Inc. Other product names mentioned herein may be trademarks and/or registered trademarks of their respective companies. Specifications and descriptions subject to change without notice.