Network Working Group
Request for Comments: 2663
Category: Informational

P. Srisuresh
M. Holdrege
Lucent Technologies

August 1999

IP Network Address Translator (NAT) Terminology and 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.

Preface

The motivation behind this document is to provide clarity to the terms used in conjunction with Network Address Translators. The term "Network Address Translator" means different things in different contexts. The intent of this document is to define the various flavors of NAT and standardize the meaning of terms used.

The authors listed are editors for this document and owe the content to contributions from members of the working group. Large chunks of the document titled, "IP Network Address Translator (NAT)" were extracted almost as is, to form the initial basis for this document. The editors would like to thank the authors Pyda Srisuresh and Kjeld Egevang for the same. The editors would like to thank Praveen Akkiraju for his contributions in describing NAT deployment scenarios. The editors would also like to thank the IESG members Scott Bradner, Vern Paxson and Thomas Narten for their detailed review of the document and adding clarity to the text.

Abstract

Network Address Translation is a method by which IP addresses are mapped from one realm to another, in an attempt to provide transparent routing to hosts. Traditionally, NAT devices are used to connect an isolated address realm with private unregistered addresses to an external realm with globally unique registered addresses. This document attempts to describe the operation of NAT devices and the associated considerations in general, and to define the terminology used to identify various flavors of NAT.

1. Introduction and Overview

The need for IP Address translation arises when a network's internal IP addresses cannot be used outside the network either because they are invalid for use outside, or because the internal addressing must be kept private from the external network.

Address translation allows (in many cases, except as noted in sections 8 and 9) hosts in a private network to transparently communicate with destinations on an external network and vice versa. There are a variety of flavors of NAT and terms to match them. This document attempts to define the terminology used and to identify various flavors of NAT. The document also attempts to describe other considerations applicable to NAT devices in general.

Note, however, this document is not intended to describe the operations of individual NAT variations or the applicability of NAT devices.

NAT devices attempt to provide a transparent routing solution to end hosts trying to communicate from disparate address realms. This is achieved by modifying end node addresses en-route and maintaining state for these updates so that datagrams pertaining to a session are routed to the right end-node in either realm. This solution only works when the applications do not use the IP addresses as part of the protocol itself. For example, identifying endpoints using DNS names rather than addresses makes applications less dependent of the actual addresses that NAT chooses and avoids the need to also translate payload contents when NAT changes an IP address.

The NAT function cannot by itself support all applications transparently and often must co-exist with application level gateways (ALGs) for this reason. People looking to deploy NAT based solutions need to determine their application requirements first and assess the NAT extensions (i.e., ALGs) necessary to provide application transparency for their environment.

IPsec techniques which are intended to preserve the Endpoint addresses of an IP packet will not work with NAT enroute for most applications in practice. Techniques such as AH and ESP protect the contents of the IP headers (including the source and destination addresses) from modification. Yet, NAT's fundamental role is to alter the addresses in the IP header of a packet.

2. Terminology and concepts used

Terms most frequently used in the context of NAT are defined here for reference.

2.1. Address realm or realm

An address realm is a network domain in which the network addresses are uniquely assigned to entities such that datagrams can be routed to them. Routing protocols used within the network domain are responsible for finding routes to entities given their network addresses. Note that this document is limited to describing NAT in IPv4 environment and does not address the use of NAT in other types of environment. (e.g. IPv6 environments)

2.2. Transparent routing

The term "transparent routing" is used throughout the document to identify the routing functionality that a NAT device provides. This is different from the routing functionality provided by a traditional router device in that a traditional router routes packets within a single address realm.

Transparent routing refers to routing a datagram between disparate address realms, by modifying address contents in the IP header to be valid in the address realm into which the datagram is routed. Section 3.2 has a detailed description of transparent routing.

2.3. Session flow vs. Packet flow

Connection or session flows are different from packet flows. A session flow indicates the direction in which the session was initiated with reference to a network interface. Packet flow is the direction in which the packet has traveled with reference to a network interface. Take for example, an outbound telnet session. The telnet session consists of packet flows in both inbound and outbound directions. Outbound telnet packets carry terminal keystrokes and inbound telnet packets carry screen displays from the telnet server.

For purposes of discussion in this document, a session is defined as the set of traffic that is managed as a unit for translation. TCP/UDP sessions are uniquely identified by the tuple of (source IP address, source TCP/UDP port, target IP address, target TCP/UDP port). ICMP query sessions are identified by the tuple of (source IP address, ICMP query ID, target IP address). All other sessions are characterized by the tuple of (source IP address, target IP address, IP protocol).

Address translations performed by NAT are session based and would include translation of incoming as well as outgoing packets belonging to that session. Session direction is identified by the direction of the first packet of that session (see sec 2.5).

Note, there is no guarantee that the idea of a session, determined as above by NAT, will coincide with the application's idea of a session. An application might view a bundle of sessions (as viewed by NAT) as a single session and might not even view its communication with its peers as a session. Not all applications are guaranteed to work across realms, even with an ALG (defined below in section 2.9) enroute.

2.4. TU ports, Server ports, Client ports

For the reminder of this document, we will refer TCP/UDP ports associated with an IP address simply as "TU ports".

For most TCP/IP hosts, TU port range 0-1023 is used by servers listening for incoming connections. Clients trying to initiate a connection typically select a source TU port in the range of 1024-65535. However, this convention is not universal and not always followed. Some client stations initiate connections using a source TU port number in the range of 0-1023, and there are servers listening on TU port numbers in the range of 1024-65535.

A list of assigned TU port services may be found in RFC 1700 [Ref 2].

2.5. Start of session for TCP, UDP and others

The first packet of every TCP session tries to establish a session and contains connection startup information. The first packet of a TCP session may be recognized by the presence of SYN bit and absence of ACK bit in the TCP flags. All TCP packets, with the exception of the first packet, must have the ACK bit set.

However, there is no deterministic way of recognizing the start of a UDP based session or any non-TCP session. A heuristic approach would be to assume the first packet with hitherto non-existent session parameters (as defined in section 2.3) as constituting the start of new session.

2.6. End of session for TCP, UDP and others

The end of a TCP session is detected when FIN is acknowledged by both halves of the session or when either half receives a segment with the RST bit in TCP flags field. However, because it is impossible for a NAT device to know whether the packets it sees will actually be delivered to the destination (they may be dropped between the NAT device and the destination), the NAT device cannot safely assume that the segments containing FINs or SYNs will be the last packets of the session (i.e., there could be retransmissions). Consequently, a session can be assumed to have been terminated only after a period of

4 minutes subsequent to this detection. The need for this extended wait period is described in RFC 793 [Ref 7], which suggests a TIME-WAIT duration of 2 * MSL (Maximum Segment Lifetime) or 4 minutes.

Note that it is also possible for a TCP connection to terminate without the NAT device becoming aware of the event (e.g., in the case where one or both peers reboot). Consequently, garbage collection is necessary on NAT devices to clean up unused state about TCP sessions that no longer exist. However, it is not possible in the general case to distinguish between connections that have been idle for an extended period of time from those that no longer exist. In the case of UDP-based sessions, there is no single way to determine when a session ends, since UDP-based protocols are application specific.

Many heuristic approaches are used to terminate sessions. You can make the assumption that TCP sessions that have not been used for say, 24 hours, and non-TCP sessions that have not been used for a couple of minutes, are terminated. Often this assumption works, but sometimes it doesn’t. These idle period session timeouts vary a great deal both from application to application and for different sessions of the same application. Consequently, session timeouts must be configurable. Even so, there is no guarantee that a satisfactory value can be found. Further, as stated in section 2.3, there is no guarantee that NAT’s view of session termination will coincide with that of the application.

Another way to handle session terminations is to timestamp entries and keep them as long as possible and retire the longest idle session when it becomes necessary.

2.7. Public/Global/External network

A Global or Public Network is an address realm with unique network addresses assigned by Internet Assigned Numbers Authority (IANA) or an equivalent address registry. This network is also referred as External network during NAT discussions.

2.8. Private/Local network

A private network is an address realm independent of external network addresses. Private network may also be referred alternately as Local Network. Transparent routing between hosts in private realm and external realm is facilitated by a NAT router.

RFC 1918 [Ref 1] has recommendations on address space allocation for private networks. Internet Assigned Numbers Authority (IANA) has three blocks of IP address space, namely 10/8, 172.16/12, and 192.168/16 set aside for private internets. In pre-CIDR notation, the

first block is nothing but a single class A network number, while the second block is a set of 16 contiguous class B networks, and the third block is a set of 256 contiguous class C networks.

An organization that decides to use IP addresses in the address space defined above can do so without coordination with IANA or any other Internet registry such as APNIC, RIPE and ARIN. The address space can thus be used privately by many independent organizations at the same time. However, if those independent organizations later decide they wish to communicate with each other or the public Internet, they will either have to renumber their networks or enable NAT on their border routers.

2.9. Application Level gateway (ALG)

Not all applications lend themselves easily to translation by NAT devices; especially those that include IP addresses and TCP/UDP ports in the payload. Application Level Gateways (ALGs) are application specific translation agents that allow an application on a host in one address realm to connect to its counterpart running on a host in different realm transparently. An ALG may interact with NAT to set up state, use NAT state information, modify application specific payload and perform whatever else is necessary to get the application running across disparate address realms.

ALGs may not always utilize NAT state information. They may glean application payload and simply notify NAT to add additional state information in some cases. ALGs are similar to Proxies, in that, both ALGs and proxies facilitate Application specific communication between clients and servers. Proxies use a special protocol to communicate with proxy clients and relay client data to servers and vice versa. Unlike Proxies, ALGs do not use a special protocol to communicate with application clients and do not require changes to application clients.

3. What is NAT?

Network Address Translation is a method by which IP addresses are mapped from one address realm to another, providing transparent routing to end hosts. There are many variations of address translation that lend themselves to different applications. However, all flavors of NAT devices should share the following characteristics.

a) Transparent Address assignment.
b) Transparent routing through address translation. (routing here refers to forwarding packets, and not exchanging routing information)
c) ICMP error packet payload translation.

Below is a diagram illustrating a scenario in which NAT is enabled on a stub domain border router, connected to the Internet through a regional router made available by a service provider.

```
    \ | /                  .                               /
+---------------+  WAN     .           +-----------------+/
|Regional Router|----------------------|Stub Router w/NAT|---
+---------------+          .           +-----------------+\
                           .                      |        \
                           .                      |  LAN
                           .               ---------------
                      Stub border
```

Figure 1: A typical NAT operation scenario

3.1. Transparent Address Assignment

NAT binds addresses in private network with addresses in global network and vice versa to provide transparent routing for the datagrams traversing between address realms. The binding in some cases may extend to transport level identifiers (such as TCP/UDP ports). Address binding is done at the start of a session. The following sub-sections describe two types of address assignments.

3.1.1. Static Address assignment

In the case of static address assignment, there is one-to-one address mapping for hosts between a private network address and an external network address for the lifetime of NAT operation. Static address assignment ensures that NAT does not have to administer address management with session flows.

3.1.2. Dynamic Address assignment

In this case, external addresses are assigned to private network hosts or vice versa, dynamically based on usage requirements and session flow determined heuristically by NAT. When the last session using an address binding is terminated, NAT would free the binding so that the global address could be recycled for later use. The exact nature of address assignment is specific to individual NAT implementations.

3.2. Transparent routing

A NAT router sits at the border between two address realms and translates addresses in IP headers so that when the packet leaves one realm and enters another, it can be routed properly. Because NAT devices have connections to multiple address realms, they must be careful to not improperly propagate information (e.g., via routing protocols) about networks from one address realm into another, where such an advertisement would be deemed unacceptable.

There are three phases to Address translation, as follows. Together these phases result in creation, maintenance and termination of state for sessions passing through NAT devices.

3.2.1. Address binding

Address binding is the phase in which a local node IP address is associated with an external address or vice versa, for purposes of translation. Address binding is fixed with static address assignments and is dynamic at session startup time with dynamic address assignments. Once the binding between two addresses is in place, all subsequent sessions originating from or to this host will use the same binding for session based packet translation.

New address bindings are made at the start of a new session, if such an address binding didn’t already exist. Once a local address is bound to an external address, all subsequent sessions originating from the same local address or directed to the same local address will use the same binding.

The start of each new session will result in the creation of a state to facilitate translation of datagrams pertaining to the session. There can be many simultaneous sessions originating from the same host, based on a single address binding.

3.2.2. Address lookup and translation

Once a state is established for a session, all packets belonging to the session will be subject to address lookup (and transport identifier lookup, in some cases) and translation.

Address or transport identifier translation for a datagram will result in the datagram forwarding from the origin address realm to the destination address realm with network addresses appropriately updated.

3.2.3. Address unbinding

Address unbinding is the phase in which a private address is no longer associated with a global address for purposes of translation. NAT will perform address unbinding when it believes that the last session using an address binding has terminated. Refer section 2.6 for some heuristic ways to handle session terminations.

3.3. ICMP error packet translation

All ICMP error messages (with the exception of Redirect message type) will need to be modified, when passed through NAT. The ICMP error message types needing NAT modification would include Destination-Unreachable, Source-Quench, Time-Exceeded and Parameter-Problem. NAT should not attempt to modify a Redirect message type.

Changes to ICMP error message will include changes to the original IP packet (or portions thereof) embedded in the payload of the ICMP error message. In order for NAT to be completely transparent to end hosts, the IP address of the IP header embedded in the payload of the ICMP packet must be modified, the checksum field of the same IP header must correspondingly be modified, and the accompanying transport header. The ICMP header checksum must also be modified to reflect changes made to the IP and transport headers in the payload. Furthermore, the normal IP header must also be modified.

4.0. Various flavors of NAT

There are many variations of address translation that lend themselves to different applications. NAT flavors listed in the following sub-sections are by no means exhaustive, but they do capture the significant differences that abound.

The following diagram will be used as a base model to illustrate NAT flavors. Host-A, with address Addr-A is located in a private realm, represented by the network N-Pri. N-Pri is isolated from external network through a NAT router. Host-X, with address Addr-X is located in an external realm, represented by the network N-Ext. NAT router with two interfaces, each attached to one of the realms provides transparent routing between the two realms. The interface to the external realm is assigned an address of Addr-Nx and the interface to private realm is assigned an address of Addr-Np. Further, it may be understood that addresses Addr-A and Addr-Np correspond to N-Pri network and the addresses Addr-X and Addr-Nx correspond to N-Ext network.

```
                                 _________________
                                (                 )
                               (   External       )    +--+
                              (  Address Realm     )-- |__|
                               (     (N-Ext)      )   /____\
                                (_________________)    Host-X
                                        |              (Addr-X)
                                        |(Addr-Nx)
                          +--------------+
                          |              |
                          |  NAT router  |
                          |              |
                          +--------------+
                            |(Addr-Np)
                            |
                     ----------------
                    (                )
       +--+        (     Private      )
       |__|------(    Address Realm   )
      /____\       (     (N-pri)      )
      Host-A        (_________________)
      (Addr-A)
```

Figure 2: A base model to illustrate NAT terms.

4.1. Traditional NAT (or) Outbound NAT

Traditional NAT would allow hosts within a private network to transparently access hosts in the external network, in most cases. In a traditional NAT, sessions are uni-directional, outbound from the private network. This is in contrast with Bi-directional NAT, which permits sessions in both inbound and outbound directions. A detailed description of Bi-directional NAT may be found in section 4.2.

The following is a description of the properties of realms supported by traditional NAT. IP addresses of hosts in external network are unique and valid in external as well as private networks. However, the addresses of hosts in private network are unique only within the private network and may not be valid in the external network. In other words, NAT would not advertise private networks to the external realm. But, networks from the external realm may be advertised within the private network. The addresses used within private network must not overlap with the external addresses. Any given address must either be a private address or an external address; not both.

A traditional NAT router in figure 2 would allow Host-A to initiate sessions to Host-X, but not the other way around. Also, N-Ext is routable from within N-Pri, whereas N-Pri may not be routable from N-Ext.

Traditional NAT is primarily used by sites using private addresses that wish to allow outbound sessions from their site.

There are two variations to traditional NAT, namely Basic NAT and NAPT (Network Address Port Translation). These are discussed in the following sub-sections.

4.1.1. Basic NAT

With Basic NAT, a block of external addresses are set aside for translating addresses of hosts in a private domain as they originate sessions to the external domain. For packets outbound from the private network, the source IP address and related fields such as IP, TCP, UDP and ICMP header checksums are translated. For inbound packets, the destination IP address and the checksums as listed above are translated.

A Basic NAT router in figure 2 may be configured to translate N-Pri into a block of external addresses, say Addr-i through Addr-n, selected from the external network N-Ext.

4.1.2. Network Address Port Translation (NAPT)

NAPT extends the notion of translation one step further by also translating transport identifier (e.g., TCP and UDP port numbers, ICMP query identifiers). This allows the transport identifiers of a number of private hosts to be multiplexed into the transport identifiers of a single external address. NAPT allows a set of hosts to share a single external address. Note that NAPT can be combined with Basic NAT so that a pool of external addresses are used in conjunction with port translation.

For packets outbound from the private network, NAPT would translate the source IP address, source transport identifier and related fields such as IP, TCP, UDP and ICMP header checksums. Transport identifier can be one of TCP/UDP port or ICMP query ID. For inbound packets, the destination IP address, destination transport identifier and the IP and transport header checksums are translated.

A NAPT router in figure 2 may be configured to translate sessions originated from N-Pri into a single external address, say Addr-i.

Very often, the external interface address Addr-Nx of NAPT router is used as the address to map N-Pri to.

4.2. Bi-directional NAT (or) Two-Way NAT

With a Bi-directional NAT, sessions can be initiated from hosts in the public network as well as the private network. Private network addresses are bound to globally unique addresses, statically or dynamically as connections are established in either direction. The name space (i.e., their Fully Qualified Domain Names) between hosts in private and external networks is assumed to be end-to-end unique. Hosts in external realm access private realm hosts by using DNS for address resolution. A DNS-ALG must be employed in conjunction with Bi-Directional NAT to facilitate name to address mapping. Specifically, the DNS-ALG must be capable of translating private realm addresses in DNS Queries and responses into their external realm address bindings, and vice versa, as DNS packets traverse between private and external realms.

The address space requirements outlined for traditional NAT routers are applicable here as well.

A Bi-directional NAT router in figure 2 would allow Host-A to initiate sessions to Host-X, and Host-X to initiate sessions to Host-A. Just as with traditional NAT, N-Ext is routable from within N-Pri, but N-Pri may not be routable from N-Ext.

4.3. Twice NAT

Twice NAT is a variation of NAT in that both the source and destination addresses are modified by NAT as a datagram crosses address realms. This is in contrast to Traditional-NAT and Bi-Directional NAT, where only one of the addresses (either source or destination) is translated. Note, there is no such term as ’Once-NAT’.

Twice NAT is necessary when private and external realms have address collisions. The most common case where this would happen is when a site had (improperly) numbered its internal nodes using public addresses that have been assigned to another organization. Alternatively, a site may have changed from one provider to another, but chosen to keep (internally) the addresses it had been assigned by the first provider. That provider might then later reassign those addresses to someone else. The key issue in such cases is that the address of the host in the external realm may have been assigned the

same address as a host within the local site. If that address were to appear in a packet, it would be forwarded to the internal node rather than through the NAT device to the external realm. Twice-NAT attempts to bridge these realms by translating both source and destination address of an IP packet, as the packet transitions realms.

Twice-NAT works as follows. When Host-A wishes to initiate a session to Host-X, it issues a DNS query for Host-X. A DNS-ALG intercepts the DNS query, and in the response returned to Host-A the DNS-ALG replaces the address for Host-X with one that is properly routable in the local site (say Host-XPRIME). Host A then initiates communication with Host-XPRIME. When the packets traverse the NAT device, the source IP address is translated (as in the case of traditional NAT) and the destination address is translated to Host-X. A similar translation is performed on return packets coming from Host-X.

The following is a description of the properties of realms supported by Twice-NAT. Network address of hosts in external network are unique in external networks, but not within private network. Likewise, the network address of hosts in private network are unique only within the private network. In other words, the address space used in private network to locate hosts in private and public networks is unrelated to the address space used in public network to locate hosts in private and public networks. Twice NAT would not be allowed to advertise local networks to the external network or vice versa.

A Twice NAT router in figure 2 would allow Host-A to initiate sessions to Host-X, and Host-X to initiate sessions to Host-A. However, N-Ext (or a subset of N-Ext) is not routable from within N-Pri, and N-Pri is not routable from N-Ext.

Twice NAT is typically used when address space used in a Private network overlaps with addresses used in the Public space. For example, say a private site uses the 200.200.200.0/24 address space which is officially assigned to another site in the public internet. Host_A (200.200.200.1) in Private space seeks to connect to Host_X (200.200.200.100) in Public space. In order to make this connection work, Host_X’s address is mapped to a different address for Host_A and vice versa. The twice NAT located at the Private site border may be configured as follows:

```
Private to Public : 200.200.200.0/24 -> 138.76.28.0/24
Public to Private : 200.200.200.0/24 -> 172.16.1.0/24

Datagram flow  : Host_A(Private) ->  Host_X(Public)

a) Within private network

   DA: 172.16.1.100      SA: 200.200.200.1

b) After twice-NAT translation

  DA: 200.200.200.100    SA: 138.76.28.1

Datagram flow Host_X (Public) -> Host_A (Private)

a) Within Public network

   DA: 138.76.28.1       SA: 200.200.200.100

b) After twice-NAT translation, in private network

   SA: 200.200.200.1     DA: 172.16.1.100
```

4.4. Multihomed NAT

There are limitations to using NAT. For example, requests and responses pertaining to a session must be routed via the same NAT router, as a NAT router maintains state information for sessions established through it. For this reason, it is often suggested that NAT routers be operated on a border router unique to a stub domain, where all IP packets are either originated from the domain or destined to the domain. However, such a configuration would turn a NAT router into a single point of failure.

In order for a private network to ensure that connectivity with external networks is retained even as one of the NAT links fail, it is often desirable to multihome the private network to same or multiple service providers with multiple connections from the private domain, be it from same or different NAT boxes.

For example, a private network could have links to two different providers and the sessions from private hosts could flow through the NAT router with the best metric for a destination. When one of NAT routers fail, the other could route traffic for all connections.

Multiple NAT boxes or multiple links on the same NAT box, sharing the same NAT configuration can provide fail-safe backup for each other. In such a case, it is necessary for backup NAT device to exchange state information so that a backup NAT can take on session load transparently when the primary NAT fails. NAT backup becomes simpler, when configuration is based on static maps.

5.0. Realm Specific IP (RSIP)

"Realm Specific IP" (RSIP) is used to characterize the functionality of a realm-aware host in a private realm, which assumes realm-specific IP address to communicate with hosts in private or external realm.

A "Realm Specific IP Client" (RSIP client) is a host in a private network that adopts an address in an external realm when connecting to hosts in that realm to pursue end-to-end communication. Packets generated by hosts on either end in such a setup would be based on addresses that are end-to-end unique in the external realm and do not require translation by an intermediary process.

A "Realm Specific IP Server" (RSIP server) is a node resident on both private and external realms, that can facilitate routing of external realm packets within a private realm. These packets may either have been originated by an RSIP client or directed to an RSIP-client. RSIP-Server may also be the same node that assigns external realm addresses to RSIP-Clients.

There are two variations to RSIP, namely Realm-specific Address IP (RSA-IP) and Realm-Specific Address and Port IP (RSAP-IP). These variations are discussed in the following sub-sections.

5.1. Realm Specific Address IP (RSA-IP)

A Realm Specific Address IP (RSA-IP) client adopts an IP address from the external address space when connecting to a host in external realm. Once an RSA-IP client assumes an external address, no other host in private or external domain can assume the same address, until that address is released by the RSA-IP client.

The following is a discussion of routing alternatives that may be pursued for the end-to-end RSA-IP packets within private realm. One approach would be to tunnel the packet to the destination. The outer header can be translated by NAT as normal without affecting the addresses used in the internal header. Another approach would be to set up a bi-directional tunnel between the RSA-IP Client and the border router straddling the two address realms. Packets to and from the client would be tunneled, but packets would be forwarded as

normal between the border router and the remote destination. Note, the tunnel from the client TO the border router may not be necessary. You might be able to just forward the packet directly. This should work so long as your internal network isn't filtering packets based on source addresses (which in this case would be external addresses).

As an example, Host-A in figure 2 above, could assume an address Addr-k from the external realm and act as RSA-IP-Client to allow end-to-end sessions between Addr-k and Addr-X. Traversal of end-to-end packets within private realm may be illustrated as follows:

First method, using NAT router enroute to translate:
==========

```
Host-A               NAT router               Host-X
------               -----------              ------

<Outer IP header, with
src=Addr-A, Dest=Addr-X>,
embedding
<End-to-end packet, with
src=Addr-k, Dest=Addr-X>
----------------------------->

                     <Outer IP header, with
                     src=Addr-k, Dest=Addr-X>,
                     embedding
                     <End-to-end packet, with
                     src=Addr-k,  Dest=Addr-X>
                     ---------------------------->

                          .
                          .
                          .

                                        <Outer IP header, with
                                        src=Addr-X, Dest=Addr-k>,
                                        embedding
                                        <End-to-end packet, with
                                        src=Addr-X, Dest=Addr-k>
                                  <---------------------------------

                     <Outer IP header, with
                     src=Addr-X, Dest=Addr-A>,
                     embedding <End-to-end packet,
                     with src=Addr-X, Dest=Addr-k>
             <-------------------------------------
```

```
Second method, using a tunnel within private realm:
===================================================

Host-A                NAT router               Host-X
------                -----------              ------

<Outer IP header, with
src=Addr-A, Dest=Addr-Np>,
embedding
<End-to-end packet, with
src=Addr-k, Dest=Addr-X>
----------------------------->

                      <End-to-end packet, with
                      src=Addr-k, Dest=Addr-X>
                      ------------------------------->

                           .
                           .
                           .

                                       <End-to-end packet, with
                                       src=Addr-X, Dest=Addr-k>
                                <--------------------------------

                      <Outer IP header, with
                      src=Addr-Np, Dest=Addr-A>,
                      embedding <End-to-end packet,
                      with src=Addr-X, Dest=Addr-k>
                 <----------------------------------
```

There may be other approaches to pursue.

An RSA-IP-Client has the following characteristics. The collective set of operations performed by an RSA-IP-Client may be termed "RSA-IP".

1. Aware of the realm to which its peer nodes belong.

2. Assumes an address from external realm when communicating with hosts in that realm. Such an address may be assigned statically or obtained dynamically (through a yet-to-be-defined protocol) from a node capable of assigning addresses from external realm. RSA-IP-Server could be the node coordinating external realm address assignment.

3. Route packets to external hosts using an approach amenable to RSA-IP-Server. In all cases, RSA-IP-Client will likely need to act as a tunnel end-point, capable of encapsulating end-to-end packets while forwarding and decapsulating in the return path.

"Realm Specific Address IP Server" (RSA-IP server) is a node resident on both private and external realms, that facilitates routing of external realm packets specific to RSA-IP clients inside a private realm. An RSA-IP-Server may be described as having the following characteristics.

1. May be configured to assign addresses from external realm to RSA-IP-Clients, either statically or dynamically.

2. Must be a router resident on both the private and external address realms.

3. Must be able to provide a mechanism to route external realm packets within private realm. Of the two approaches described, the first approach requires RSA-IP-Server to be a NAT router providing transparent routing for the outer header. This approach requires the external peer to be a tunnel end-point.

   With the second approach, an RSA-IP-Server could be any router (including a NAT router) that can be a tunnel end-point with RSA-IP-Clients. It would detunnel end-to-end packets outbound from RSA-IP-Clients and forward to external hosts. On the return path, it would locate RSA-IP-Client tunnel, based on the destination address of the end-to-end packet and encapsulate the packet in a tunnel to forward to RSA-IP-Client.

RSA-IP-Clients may pursue any of the IPsec techniques, namely transport or tunnel mode Authentication and confidentiality using AH and ESP headers on the embedded packets. Any of the tunneling techniques may be adapted for encapsulation between RSA-IP-Client and RSA-IP-Server or between RSA-IP-Client and external host. For example, IPsec tunnel mode encapsulation is a valid type of encapsulation that ensures IPsec authentication and confidentiality for the embedded end-to-end packets.

5.2 Realm Specific Address and port IP (RSAP-IP)

Realm Specific Address and port IP (RSAP-IP) is a variation of RSIP in that multiple private hosts use a single external address, multiplexing on transport IDentifiers (i.e., TCP/UDP port numbers and ICMP Query IDs).

"RSAP-IP-Client" may be defined similar to RSA-IP-Client with the variation that RSAP-IP-Client assumes a tuple of (external address, transport Identifier) when connecting to hosts in external realm to pursue end-to-end communication. As such, communication with external nodes for an RSAP-IP-Client may be limited to TCP, UDP and ICMP sessions.

"RSAP-IP-Server" is similar to RSA-IP-Server in that it facilitates routing of external realm packets specific to RSAP-IP clients inside a private realm. Typically, an RSAP-IP-Server would also be the one to assign transport tuples to RSAP-IP-Clients.

A NAPT router enroute could serve as RSAP-IP-Server, when the outer encapsulation is TCP/UDP based and is addressed between the RSAP-IP-Client and external peer. This approach requires the external peer to be the end-point of TCP/UDP based tunnel. Using this approach, RSAP-IP-Clients may pursue any of the IPsec techniques, namely transport or tunnel mode authentication and confidentiality using AH and ESP headers on the embedded packets. Note however, IPsec tunnel mode is not a valid type of encapsulation, as a NAPT router cannot provide routing transparency to AH and ESP protocols.

Alternately, packets may be tunneled between RSAP-IP-Client and RSAP-IP-Server such that RSAP-IP-Server would detunnel packets outbound from RSAP-IP-Clients and forward to external hosts. On the return path, RSAP-IP-Server would locate RSAP-IP-Client tunnel, based on the tuple of (destination address, transport Identifier) and encapsulate the original packet within a tunnel to forward to RSAP-IP-Client. With this approach, there is no limitation on the tunneling technique employed between RSAP-IP-Client and RSAP-IP-Server. However, there are limitations to applying IPsec based security on end-to-end packets. Transport mode based authentication and integrity may be attained. But, confidentiality cannot be permitted because RSAP-IP-Server must be able to examine the destination transport Identifier in order to identify the RSAP-IP-tunnel to forward inbound packets to. For this reason, only the transport mode TCP, UDP and ICMP packets protected by AH and ESP-authentication can traverse a RSAP-IP-Server using this approach.

As an example, say Host-A in figure 2 above, obtains a tuple of (Addr-Nx, TCP port T-Nx) from NAPT router to act as RSAP-IP-Client to initiate end-to-end TCP sessions with Host-X. Traversal of end-to-end packets within private realm may be illustrated as follows. In the first method, outer layer of the outgoing packet from Host-A uses (private address Addr-A, source port T-Na) as source tuple to communicate with Host-X. NAPT router enroute translates this tuple into (Addr-Nx, Port T-Nxa). This translation is independent of RSAP-IP-Client tuple parameters used in the embedded packet.

```
First method, using NAPT router enroute to translate:
=====================================================

Host-A               NAPT router              Host-X
------               -----------              ------

<Outer TCP/UDP packet, with
src=Addr-A, Src Port=T-Na,
Dest=Addr-X>,
embedding
<End-to-end packet, with
src=Addr-Nx, Src Port=T-Nx, Dest=Addr-X>
----------------------------->

                     <Outer TCP/UDP packet, with
                     src=Addr-Nx, Src Port=T-Nxa,
                     Dest=Addr-X>,
                     embedding
                     <End-to-end packet, with
                     src=Addr-Nx, Src Port=T-Nx, Dest=Addr-X>
                     ---------------------------------------->

                          .
                          .
                          .

                                     <Outer TCP/UDP packet with
                                     src=Addr-X, Dest=Addr-Nx,
                                     Dest Port=T-Nxa>,
                                     embedding
                                     <End-to-end packet, with
                                     src=Addr-X, Dest=Addr-Nx,
                                     Dest Port=T-Nx>
                               <----------------------------------

                     <Outer TCP/UDP packet, with
                     src=Addr-X, Dest=Addr-A,
                     Dest Port=T-Na>,
                     embedding
                     <End-to-end packet, with
                     src=Addr-X, Dest=Addr-Nx,
                     Dest Port=T-Nx>
               <-----------------------------------
```

```
Second method, using a tunnel within private realm:
====================================================

Host-A                NAPT router              Host-X
------                -----------              ------

<Outer IP header, with
src=Addr-A, Dest=Addr-Np>,
embedding
<End-to-end packet, with
src=Addr-Nx, Src Port=T-Nx,
Dest=Addr-X>
----------------------------->

                      <End-to-end packet, with
                      src=Addr-Nx, Src Port=T-Nx,
                      Dest=Addr-X>
                      -------------------------------->

                           .
                           .
                           .

                                       <End-to-end packet, with
                                       src=Addr-X, Dest=Addr-Nx,
                                       Dest Port=T-Nx>
                                 <----------------------------------

                      <Outer IP header, with
                      src=Addr-Np, Dest=Addr-A>,
                      embedding
                      <End-to-end packet, with
                      src=Addr-X, Dest=Addr-Nx,
                      Dest Port=T-Nx>
                <----------------------------------
```

6.0. Private Networks and Tunnels

Let us consider the case where your private network is connected to the external world via tunnels. In such a case, tunnel encapsulated traffic may or may not contain translated packets depending upon the characteristics of address realms a tunnel is bridging.

The following subsections discuss two scenarios where tunnels are used (a) in conjunction with Address translation, and (b) without translation.

6.1. Tunneling translated packets

All variations of address translations discussed in the previous section can be applicable to direct connected links as well as tunnels and virtual private networks (VPNs).

For example, a private network connected to a business partner through a VPN could employ traditional NAT to communicate with the partner. Likewise, it is possible to employ twice NAT, if the partner's address space overlapped with the private network. There could be a NAT device on one end of the tunnel or on both ends of the tunnel. In all cases, traffic across the VPN can be encrypted for security purposes. Security here refers to security for traffic across VPNs alone. End-to-end security requires trusting NAT devices within private network.

6.2. Backbone partitioned private Networks

There are many instances where a private network (such as a corporate network) is spread over different locations and use public backbone for communications between those locations. In such cases, it is not desirable to do address translation, both because large numbers of hosts may want to communicate across the backbone, thus requiring large address tables, and because there will be more applications that depend on configured addresses, as opposed to going to a name server. We call such a private network a backbone-partitioned private network.

Backbone-partitioned stubs should behave as though they were a non-partitioned stub. That is, the routers in all partitions should maintain routes to the local address spaces of all partitions. Of course, the (public) backbones do not maintain routes to any local addresses. Therefore, the border routers must tunnel (using VPNs) through the backbones using encapsulation. To do this, each NAT box will set aside a global address for tunneling.

When a NAT box x in stub partition X wishes to deliver a packet to stub partition Y, it will encapsulate the packet in an IP header with destination address set to the global address of NAT box y that has been reserved for encapsulation. When NAT box y receives a packet with that destination address, it decapsulates the IP header and routes the packet internally. Note, there is no address translation in the process; merely transfer of private network packets over an external network tunnel backbone.

7.0. NAT operational characteristics

NAT devices are application unaware in that the translations are limited to IP/TCP/UDP/ICMP headers and ICMP error messages only. NAT devices do not change the payload of the packets, as payloads tend to be application specific.

NAT devices (without the inclusion of ALGs) do not examine or modify transport payload. For this reason, NAT devices are transparent to applications in many cases. There are two areas, however, where NAT devices often cause difficulties: 1) when an application payload includes an IP address, and 2) when end-to-end security is needed. Note, this is not a comprehensive list.

Application layer security techniques that do not make use of or depend on IP addresses will work correctly in the presence of NAT (e.g., TLS, SSL and ssh). In contrast, transport layer techniques such as IPSec transport mode or the TCP MD5 Signature Option RFC 2385 [Ref 17] do not.

In IPSec transport mode, both AH and ESP have an integrity check covering the entire payload. When the payload is TCP or UDP, the TCP/UDP checksum is covered by the integrity check. When a NAT device modifies an address the checksum is no longer valid with respect to the new address. Normally, NAT also updates the checksum, but this is ineffective when AH and ESP are used. Consequently, receivers will discard a packet either because it fails the IPSec integrity check (if the NAT device updates the checksum), or because the checksum is invalid (if the NAT device leaves the checksum unmodified).

Note that IPsec tunnel mode ESP is permissible so long as the embedded packet contents are unaffected by the outer IP header translation. Although this technique does not work in traditional NAT deployments (i.e., where hosts are unaware that NATs are present), the technique is applicable to Realm-Specific IP as described in Section 5.0.

Note also that end-to-end ESP based transport mode authentication and confidentiality are permissible for packets such as ICMP, whose IP payload content is unaffected by the outer IP header translation.

NAT devices also break fundamental assumptions by public key distribution infrastructures such as Secure DNS RFC 2535 [Ref 18] and X.509 certificates with signed public keys. In the case of Secure

DNS, each DNS RRset is signed with a key from within the zone. Moreover, the authenticity of a specific key is verified by following a chain of trust that goes all the way to the DNS root. When a DNS-ALG modifies addresses (e.g., as in the case of Twice-NAT), verification of signatures fails.

It may be of interest to note that IKE (Session key negotiation protocol) is a UDP based session layer protocol and is not protected by network based IPsec security. Only a portion of the individual payloads within IKE are protected. As a result, IKE sessions are permissible across NAT, so long as IKE payload does not contain addresses and/or transport IDs specific to one realm and not the other. Given that IKE is used to setup IPSec associations, and there are at present no known ways of making IPSec work through a NAT function, it is a future work item to take advantage of IKE through a NAT box.

One of the most popular internet applications "FTP" would not work with the definition of NAT as described. The following sub-section is devoted to describing how FTP is supported on NAT devices. FTP ALG is an integral part of most NAT implementations. Some vendors may choose to include additional ALGs to custom support other applications on the NAT device.

7.1. FTP support

"PORT" command and "PASV" response in FTP control session payload identify the IP address and TCP port that must be used for the data session it supports. The arguments to the PORT command and PASV response are an IP address and a TCP port in ASCII. An FTP ALG is required to monitor and update the FTP control session payload so that information contained in the payload is relevant to end nodes. The ALG must also update NAT with appropriate data session tuples and session orientation so that NAT could set up state information for the FTP data sessions.

Because the address and TCP port are encoded in ASCII, this may result in a change in the size of packet. For instance, 10,18,177,42,64,87 is 18 ASCII characters, whereas 193,45,228,137,64,87 is 20 ASCII characters. If the new size is same as the previous, only the TCP checksum needs adjustment as a result of change of data. If the new size is less than or greater than the previous, TCP sequence numbers must also be changed to reflect the change in length of FTP control data portion. A special table may be used by the ALG to correct the TCP sequence and acknowledge numbers. The sequence number and acknowledgement correction will need to be performed on all future packet of the connection.

8.0. NAT limitations

8.1. Applications with IP-address Content

Not All applications lend themselves easily to address translation by NAT devices. Especially, the applications that carry IP address (and TU port, in case of NAPT) inside the payload. Application Level Gateways, or ALGs must be used to perform translations on packets pertaining to such applications. ALGs may optionally utilize address (and TU port) assignments made by NAT and perform translations specific to the application. The combination of NAT functionality and ALGs will not provide end-to-end security assured by IPsec. However, tunnel mode IPsec can be accomplished with NAT router serving as tunnel end point.

SNMP is one such application with address content in payload. NAT routers would not translate IP addresses within SNMP payloads. It is not uncommon for an SNMP specific ALG to reside on a NAT router to perform SNMP MIB translations proprietary to the private network.

8.2. Applications with inter-dependent control and data sessions

NAT devices operate on the assumption that each session is independent. Session characteristics like session orientation, source and destination IP addresses, session protocol, and source and destination transport level identifiers are determined independently at the start of each new session.

However, there are applications such as H.323 that use one or more control sessions to set the characteristics of the follow-on sessions in their control session payload. Such applications require use of application specific ALGs that can interpret and translate the payload, if necessary. Payload interpretation would help NAT be prepared for the follow-on data sessions.

8.3. Debugging Considerations

NAT increases the probability of mis-addressing. For example, same local address may be bound to different global address at different times and vice versa. As a result, any traffic flow study based purely on global addresses and TU ports could be confused and might misinterpret the results.

If a host is abusing the Internet in some way (such as trying to attack another machine or even sending large amounts of junk mail or something) it is more difficult to pinpoint the source of the trouble because the IP address of the host is hidden in a NAT router.

8.4. Translation of fragmented FTP control packets

Translation of fragmented FTP control packets is tricky when the packets contain "PORT" command or response to "PASV" command. Clearly, this is a pathological case. NAT router would need to assemble the fragments together first and then translate prior to forwarding.

Yet another case would be when each character of packets containing "PORT" command or response to "PASV" is sent in a separate datagram, unfragmented. In this case, NAT would simply have to let the packets through, without translating the TCP payload. Of course, the application will fail if the payload needed to be altered. The application could still work in a few cases, where the payload contents can be valid in both realms, without modifications enroute. For example, FTP originated from a private host would still work while traversing a traditional NAT or bi-directional NAT device, so long as the FTP control session employed PASV command to establish data sessions. The reason being that the address and port number specified by FTP server in the PASV response (sent as multiple unfragmented packets) is valid to the private host, as is. The NAT device will simply view the ensuing data session (also originating from private host) as an independent TCP session.

8.5. Compute intensive

NAT is compute intensive even with the help of a clever checksum adjustment algorithm, as each data packet is subject to NAT lookup and modifications. As a result, router forwarding throughput could be slowed considerably. However, so long as the processing capacity of the NAT device exceeds line processing rate, this should not be a problem.

9.0. Security Considerations

Many people view traditional NAT router as a one-way (session) traffic filter, restricting sessions from external hosts into their machines. In addition, when address assignment in NAT router is done dynamically, that makes it harder for an attacker to point to any specific host in the NAT domain. NAT routers may be used in conjunction with firewalls to filter unwanted traffic.

If NAT devices and ALGs are not in a trusted boundary, that is a major security problem, as ALGs could snoop end user traffic payload. Session level payload could be encrypted end to end, so long as the payload does not contain IP addresses and/or transport identifiers that are valid in only one of the realms. With the exception of RSIP, end-to-end IP network level security assured by current IPsec

techniques is not attainable with NAT devices in between. One of the ends must be a NAT box. Refer section 7.0 for a discussion on why end-to-end IPsec security cannot be assured with NAT devices along the route.

The combination of NAT functionality, ALGs and firewalls will provide a transparent working environment for a private networking domain. With the exception of RSIP, end-to-end network security assured by IPsec cannot be attained for end-hosts within the private network (Refer section 5.0 for RSIP operation). In all other cases, if you want to use end-to-end IPsec, there cannot be a NAT device in the path. If we make the assumption that NAT devices are part of a trusted boundary, tunnel mode IPsec can be accomplished with NAT router (or a combination of NAT, ALGs and firewall) serving as tunnel end point.

NAT devices, when combined with ALGs, can ensure that the datagrams injected into Internet have no private addresses in headers or payload. Applications that do not meet these requirements may be dropped using firewall filters. For this reason, it is not uncommon to find NAT, ALG and firewall functions co-exist to provide security at the borders of a private network. NAT gateways can be used as tunnel end points to provide secure VPN transport of packet data across an external network domain.

Below are some additional security considerations associated with NAT routers.

1. UDP sessions are inherently unsafe. Responses to a datagram could come from an address different from the target address used by sender ([Ref 4]). As a result, an incoming UDP packet might match the outbound session of a traditional NAT router only in part (the destination address and UDP port number of the packet match, but the source address and port number may not). In such a case, there is a potential security compromise for the NAT device in permitting inbound packets with partial match. This UDP security issue is also inherent to firewalls.

   Traditional NAT implementations that do not track datagrams on a per-session basis but lump states of multiple UDP sessions using the same address binding into a single unified session could compromise the security even further. This is because, the granularity of packet matching would be further limited to just the destination address of the inbound UDP packets.

2. Multicast sessions (UDP based) are another source for security weakness for traditional-NAT routers. Once again, firewalls face the same security dilemma as the NAT routers.

Say, a host on private network initiated a multicast session. Datagram sent by the private host could trigger responses in the reverse direction from multiple external hosts. Traditional-NAT implementations that use a single state to track a multicast session cannot determine for certain if the incoming UDP packet is in response to an existing multicast session or the start of new UDP session initiated by an attacker.

3. NAT devices can be a target for attacks.

   Since NAT devices are Internet hosts they can be the target of a number of different attacks, such as SYN flood and ping flood attacks. NAT devices should employ the same sort of protection techniques as Internet-based servers do.

REFERENCES

[1] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot,G. and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

[2] Reynolds, J. and J. Postel, "Assigned Numbers", STD 2, RFC 1700, October, 1994.

[3] Braden, R., "Requirements for Internet Hosts -- Communication Layers", STD 3, RFC 1122, October 1989.

[4] Braden, R., "Requirements for Internet Hosts -- Application and Support", STD 3, RFC 1123, October 1989.

[5] Baker, F., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.

[6] Postel, J. and J. Reynolds, "File Transfer Protocol (FTP)", STD 9, RFC 959, October 1985.

[7] Postel, J., "Transmission Control Protocol (TCP) Specification", STD 7, RFC 793, September 1981.

[8] Postel, J., "Internet Control Message Protocol Specification" STD 5, RFC 792, September 1981.

[9] Postel, J., "User Datagram Protocol (UDP)", STD 6, RFC 768, August 1980.

[10] Mogul, J. and J. Postel, "Internet Standard Subnetting Procedure", STD 5, RFC 950, August 1985.

[11] Carpenter, B., Crowcroft, J. and Y. Rekhter, "IPv4 Address Behavior Today", RFC 2101, February 1997.

[12] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[13] Kent, S. and R. Atkinson, "IP Encapsulating Security Payload (ESP)", RFC 2406, November 1998.

[14] Kent, S. and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.

[15] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[16] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP", RFC 2407, November 1998.

[17] Heffernan, A., "Protection of BGP Sessions via the TCP MD5 Signature Option", RFC 2385, August 1998.

[18] Eastlake, D., "Domain Name System Security Extensions", RFC 2535, March 1999.

Authors' Addresses

Pyda Srisuresh
Lucent Technologies
4464 Willow Road
Pleasanton, CA 94588-8519
U.S.A.

Phone: (925) 737-2153
Fax: (925) 737-2110
EMail: srisuresh@lucent.com

Matt Holdrege
Lucent Technologies
1701 Harbor Bay Parkway
Alameda, CA 94502

Phone: (510) 769-6001
EMail: holdrege@lucent.com

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.