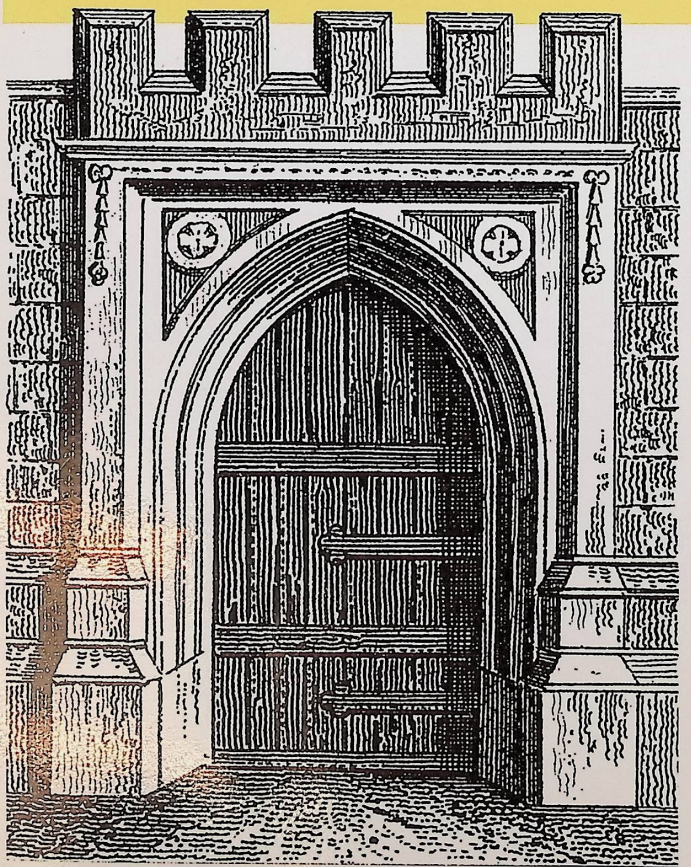
Internet and Web Security

2nd Edition

Covers Unix, Windows NT, and Linux

Building Internet

Firewalls



O'REILLY®

Elizabeth D. Zwicky, Simon Cooper & D. Brent Chapman

FreeWheel, Ex. 1112

Building Internet Firewalls

Building Internet Firewalls

Second Edition

Elizabeth D. Zwicky, Simon Cooper &
D. Brent Chapman

O'REILLY®

Beijing · Cambridge · Farnham · Köln · Paris · Sebastopol · Taipei · Tokyo

Building Internet Firewalls, Second Edition
by Elizabeth D. Zwicky, Simon Cooper, and D. Brent Chapman

Copyright © 2000 O'Reilly & Associates, Inc. All rights reserved.
Printed in the United States of America.

Published by O'Reilly & Associates, Inc., 101 Morris Street, Sebastopol, CA 95472.

Editor: Deborah Russell

Production Editor: Nancy Crumpton

Production Coordinator: Madeleine Newell

Cover Designer: Edie Freedman

Printing History:

April 1995:	First Edition.
November 1995:	Minor corrections.
June 2000:	Second Edition.

Nutshell Handbook, the Nutshell Handbook logo, and the O'Reilly logo are registered trademarks of O'Reilly & Associates, Inc. Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc., in the United States and other countries. O'Reilly & Associates, Inc. is independent of Sun Microsystems.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly & Associates, Inc. was aware of a trademark claim, the designations have been printed in caps or initial caps. The association between the image of a Gothic doorway and the topic of Internet firewalls is a trademark of O'Reilly & Associates, Inc.

While every precaution has been taken in the preparation of this book, the publisher assumes no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

Library of Congress Cataloging-in-Publication Data

Zwicky, Elizabeth D.
Building Internet firewalls / Elizabeth D. Zwicky, Simon Cooper & D. Brent Chapman.--
2nd ed. p. cm.
ISBN 1-56592-871-7
1. Computer networks--Security measures. 2. Internet (Computer network)--Security measures. I. Cooper, Simon. II. Chapman, D. Brent. III. Title.

TK5105.59.Z85 2000
005.8--dc21 00-039954

ISBN: 1-56592-871-7 [12/00]
[M]

Table of Contents

Preface ..... *xiii*

I. Network Security *1*

1. Why Internet Firewalls? *3*
What Are You Trying to Protect? *4*
What Are You Trying to Protect Against? *7*
Who Do You Trust? *16*
How Can You Protect Your Site? *17*
What Is an Internet Firewall? *21*
Religious Arguments *28*

2. Internet Services *33*
Secure Services and Safe Services *35*
The World Wide Web *35*
Electronic Mail and News *40*
File Transfer, File Sharing, and Printing *43*
Remote Access *48*
Real-Time Conferencing Services *51*
Naming and Directory Services *52*
Authentication and Auditing Services *54*
Administrative Services *55*
Databases *58*
Games *58*

3. *Security Strategies* 59
Least Privilege 59
Defense in Depth 61
Choke Point 62
Weakest Link 63
Fail-Safe Stance 64
Universal Participation 67
Diversity of Defense 68
Simplicity 70
Security Through Obscurity 71

II. *Building Firewalls* 73

4. *Packets and Protocols* 75
What Does a Packet Look Like? 75
IP 79
Protocols Above IP 85
Protocols Below IP 93
Application Layer Protocols 94
IP Version 6 94
Non-IP Protocols 96
Attacks Based on Low-Level Protocol Details 97

5. *Firewall Technologies* 102
Some Firewall Definitions 102
Packet Filtering 104
Proxy Services 110
Network Address Translation 114
Virtual Private Networks 119

6. *Firewall Architectures* 122
Single-Box Architectures 122
Screened Host Architectures 126
Screened Subnet Architectures 128
Architectures with Multiple Screened Subnets 133
Variations on Firewall Architectures 137
Terminal Servers and Modem Pools 148
Internal Firewalls 149

7. Firewall Design *157*
Define Your Needs *157*
Evaluate the Available Products *159*
Put Everything Together *162*

8. Packet Filtering *165*
What Can You Do with Packet Filtering? *166*
Configuring a Packet Filtering Router *171*
What Does the Router Do with Packets? *173*
Packet Filtering Tips and Tricks *178*
Conventions for Packet Filtering Rules *180*
Filtering by Address *183*
Filtering by Service *185*
Choosing a Packet Filtering Router *190*
Packet Filtering Implementations for General-Purpose Computers *203*
Where to Do Packet Filtering *214*
What Rules Should You Use? *216*
Putting It All Together *216*

9. Proxy Systems *224*
Why Proxying? *225*
How Proxying Works *226*
Proxy Server Terminology *231*
Proxying Without a Proxy Server *232*
Using SOCKS for Proxying *233*
Using the TIS Internet Firewall Toolkit for Proxying *237*
Using Microsoft Proxy Server *238*
What If You Can't Proxy? *239*

10. Bastion Hosts *241*
General Principles *242*
Special Kinds of Bastion Hosts *243*
Choosing a Machine *244*
Choosing a Physical Location *248*
Locating Bastion Hosts on the Network *249*
Selecting Services Provided by a Bastion Host *250*
Disabling User Accounts on Bastion Hosts *253*
Building a Bastion Host *255*
Securing the Machine *256*
Disabling Nonrequired Services *259*

Operating the Bastion Host 269
Protecting the Machine and Backups 270

11. Unix and Linux Bastion Hosts 273
Which Version of Unix? 273
Securing Unix 275
Disabling Nonrequired Services 278
Installing and Modifying Services 288
Reconfiguring for Production 291
Running a Security Audit 295

12. Windows NT and Windows 2000 Bastion Hosts 297
Approaches to Building Windows NT Bastion Hosts 297
Which Version of Windows NT? 298
Securing Windows NT 299
Disabling Nonrequired Services 301
Installing and Modifying Services 313

III. Internet Services 315

13. Internet Services and Firewalls 317
Attacks Against Internet Services 319
Evaluating the Risks of a Service 327
Analyzing Other Protocols 334
What Makes a Good Firewalled Service? 336
Choosing Security-Critical Programs 339
Controlling Unsafe Configurations 347

14. Intermediary Protocols 349
Remote Procedure Call (RPC) 349
Distributed Component Object Model (DCOM) 358
NetBIOS over TCP/IP (NetBT) 359
Common Internet File System (CIFS) and Server Message Block (SMB) 361
Common Object Request Broker Architecture (CORBA) and Internet Inter-Orb Protocol (IIOP) 365
ToolTalk 367
Transport Layer Security (TLS) and Secure Socket Layer (SSL) 368
The Generic Security Services API (GSSAPI) 373
IPsec 373

Remote Access Service (RAS) 377
Point-to-Point Tunneling Protocol (PPTP) 378
Layer 2 Transport Protocol (L2TP) 381

15. The World Wide Web 384
HTTP Server Security 385
HTTP Client Security 390
HTTP 397
Mobile Code and Web-Related Languages 406
Cache Communication Protocols 412
Push Technologies 415
RealAudio and RealVideo 417
Gopher and WAIS 419

16. Electronic Mail and News 423
Electronic Mail 423
Simple Mail Transfer Protocol (SMTP) 430
Other Mail Transfer Protocols 441
Microsoft Exchange 442
Lotus Notes and Domino 443
Post Office Protocol (POP) 445
Internet Message Access Protocol (IMAP) 448
Microsoft Messaging API (MAPI) 450
Network News Transfer Protocol (NNTP) 450

17. File Transfer, File Sharing, and Printing 454
File Transfer Protocol (FTP) 455
Trivial File Transfer Protocol (TFTP) 468
Network File System (NFS) 470
File Sharing for Microsoft Networks 479
Summary of Recommendations for File Sharing 482
Printing Protocols 483
Related Protocols 487

18. Remote Access to Hosts 488
Terminal Access (Telnet) 488
Remote Command Execution 491
Remote Graphical Interfaces 507

19. Real-Time Conferencing Services *520*
Internet Relay Chat (IRC) *520*
ICQ *523*
talk *525*
Multimedia Protocols *528*
NetMeeting *533*
Multicast and the Multicast Backbone (MBONE) *535*

20. Naming and Directory Services *539*
Domain Name System (DNS) *539*
Network Information Service (NIS) *563*
NetBIOS for TCP/IP Name Service and Windows Internet Name Service *565*
The Windows Browser *576*
Lightweight Directory Access Protocol (LDAP) *583*
Active Directory *585*
Information Lookup Services *586*

21. Authentication and Auditing Services *591*
What Is Authentication? *592*
Passwords *596*
Authentication Mechanisms *600*
Modular Authentication for Unix *604*
Kerberos *609*
NTLM Domains *615*
Remote Authentication Dial-in User Service (RADIUS) *622*
TACACS and Friends *625*
Auth and identd *627*

22. Administrative Services *630*
System Management Protocols *630*
Routing Protocols *637*
Protocols for Booting and Boot-Time Configuration *644*
ICMP and Network Diagnostics *647*
Network Time Protocol (NTP) *654*
File Synchronization *658*
Mostly Harmless Protocols *661*

23. Databases and Games 664
Databases 664
Games 678

24. Two Sample Firewalls 681
Screened Subnet Architecture 681
Merged Routers and Bastion Host Using General-Purpose Hardware 704

IV. Keeping Your Site Secure 721

25. Security Policies 723
Your Security Policy 724
Putting Together a Security Policy 731
Getting Strategic and Policy Decisions Made 734
What If You Can't Get a Security Policy? 741

26. Maintaining Firewalls 742
Housekeeping 742
Monitoring Your System 746
Keeping up to Date 758
How Long Does It Take? 762
When Should You Start Over? 762

27. Responding to Security Incidents 764
Responding to an Incident 764
What to Do After an Incident 774
Pursuing and Capturing the Intruder 775
Planning Your Response 778
Being Prepared 787

V. Appendixes 795

A. Resources 797

B. Tools 813

C. Cryptography 823

Index 849

Preface

This book is a practical guide to building your own firewall. It provides step-by-step explanations of how to design and install a firewall at your site and how to configure Internet services such as electronic mail, FTP, the World Wide Web, and others to work with a firewall. Firewalls are complex, though, and we can't boil everything down to simple rules. Too much depends on exactly what hardware, operating system, and networking you are using at your site, and what you want your users to be able to do and not do. We've tried to give you enough rules, examples, and resources here so you'll be able to do the rest on your own.

What is a firewall, and what does it do for you? A firewall is a way to restrict access between the Internet and your internal network. You typically install a firewall at the point of maximum leverage, the point where your network connects to the Internet. The existence of a firewall at your site can greatly reduce the odds that outside attackers will penetrate your internal systems and networks. The firewall can also keep your own users from compromising your systems by sending dangerous information—unencrypted passwords and sensitive data—to the outside world.

The attacks on Internet-connected systems we are seeing today are more serious and more technically complex than those in the past. To keep these attacks from compromising our systems, we need all the help we can get. Firewalls are a highly effective way of protecting sites from these attacks. For that reason, we strongly recommend you include a firewall in your site's overall Internet security plan. However, a firewall should be only one component in that plan. It's also vital that you establish a security policy, that you implement strong host security, and that you consider the use of authentication and encryption devices that work with the firewalls you install. This book will touch on each of these topics while maintaining its focus on firewalls.

Scope of This Book

This book is divided into five parts.

Part I, *Network Security*, explores the problem of Internet security and focuses on firewalls as part of an effective strategy to address that problem.

- Chapter 1, *Why Internet Firewalls?*, introduces the major risks associated with using the Internet today; discusses what to protect, and what to protect against; discusses various security models; and introduces firewalls in the context of what they can and can't do for your site's security.
- Chapter 2, *Internet Services*, outlines the services users want and need from the Internet, and summarizes the security problems posed by those services.
- Chapter 3, *Security Strategies*, outlines the basic security principles an organization needs to understand before it adopts a security policy and invests in specific security mechanisms.

Part II, *Building Firewalls*, describes how to build firewalls.

- Chapter 4, *Packets and Protocols*, describes the basic network concepts firewalls work with.
- Chapter 5, *Firewall Technologies*, explains the terms and technologies used in building firewalls.
- Chapter 6, *Firewall Architectures*, describes the major architectures used in constructing firewalls, and the situations they are best suited to.
- Chapter 7, *Firewall Design*, presents the process of designing a firewall.
- Chapter 8, *Packet Filtering*, describes how packet filtering systems work, and discusses what you can and can't accomplish with them in building a firewall.
- Chapter 9, *Proxy Systems*, describes how proxy clients and servers work, and how to use these systems in building a firewall.
- Chapter 10, *Bastion Hosts*, presents a general overview of the process of designing and building the bastion hosts used in many firewall configurations.
- Chapter 11, *Unix and Linux Bastion Hosts*, presents the details of designing and building a Unix or Linux bastion host.
- Chapter 12, *Windows NT and Windows 2000 Bastion Hosts*, presents the details of designing and building a Windows NT bastion host.

Part III, *Internet Services*, describes how to configure services in the firewall environment.

- Chapter 13, *Internet Services and Firewalls*, describes the general issues involved in selecting and configuring services in the firewall environment.

- Chapter 14, *Intermediary Protocols*, discusses basic protocols that are used by multiple services.
- Chapter 15, *The World Wide Web*, discusses the Web and related services.
- Chapter 16, *Electronic Mail and News*, discusses services used for transferring electronic mail and Usenet news.
- Chapter 17, *File Transfer, File Sharing, and Printing*, discusses the services used for moving files from one place to another.
- Chapter 18, *Remote Access to Hosts*, discusses services that allow you to use one computer from another computer.
- Chapter 19, *Real-Time Conferencing Services*, discusses services that allow people to interact with each other online.
- Chapter 20, *Naming and Directory Services*, discusses the services used to distribute information about hosts and users.
- Chapter 21, *Authentication and Auditing Services*, discusses services used to identify users before they get access to resources, to keep track of what sort of access they should have, and to keep records of who accessed what and when.
- Chapter 22, *Administrative Services*, discusses other services used to administer machines and networks.
- Chapter 23, *Databases and Games*, discusses the remaining two major classes of popular Internet services, databases and games.
- Chapter 24, *Two Sample Firewalls*, presents two sample configurations for basic firewalls.

Part IV, *Keeping Your Site Secure*, describes how to establish a security policy for your site, maintain your firewall, and handle the security problems that may occur with even the most effective firewalls.

- Chapter 25, *Security Policies*, discusses the importance of having a clear and well-understood security policy for your site, and what that policy should and should not contain. It also discusses ways of getting management and users to accept the policy.
- Chapter 26, *Maintaining Firewalls*, describes how to maintain security at your firewall over time and how to keep yourself aware of new Internet security threats and technologies.
- Chapter 27, *Responding to Security Incidents*, describes what to do when a break-in occurs, or when you suspect that your security is being breached.

Part V, *Appendixes*, consists of the following summary appendixes:

- Appendix A, *Resources*, contains a list of places you can go for further information and help with Internet security: World Wide Web pages, FTP sites, mailing lists, newsgroups, response teams, books, papers, and conferences.
- Appendix B, *Tools*, summarizes the best freely available firewall tools and how to get them.
- Appendix C, *Cryptography*, contains background information on cryptography that is useful to anyone trying to decrypt the marketing materials for security products.

Audience

Who should read this book? Although the book is aimed primarily at those who need to build firewalls, large parts of it are appropriate for everyone who is concerned about Internet security. This list tells you what sections are particularly applicable to you:

System administrators
: You should read the entire book.

Senior managers
: You should read at least Part I of the book. The chapters in Part I will introduce you to the various types of Internet threats, services, and security approaches and strategies. These chapters will also introduce you to firewalls and describe what firewalls can and cannot do to enforce Internet security. You should also read Chapter 5, which provides an overview of firewall technologies. In addition, Appendix A will tell you where to go for more information and resources.

Information technology managers and users
: You should read all of the chapters we've cited for the managers in the previous category. In addition, you should read Part III, which explains the kinds of issues that may arise at your site over time—for example, how to develop a security policy, keep up to date, and react if someone attacks your site.

Although this book provides general concepts of firewalls appropriate to any site, it focuses on "average" sites: small to large commercial or educational sites. If you are setting up a personal firewall, you may wish to read just Part I, Chapter 5, and the service chapters appropriate to the services you wish to run. If you are setting up a firewall for an extremely large site, all of the chapters will be useful to you, but you may find that you need to use additional techniques.

Platforms

To a large extent, this book is platform-independent. Because most of the information provided here consists of general principles, most of it should be applicable to you, regardless of what equipment, software, and networking you are using. The most platform-specific issue is what type of system to use as a bastion host. People have successfully built bastion hosts (which we describe in Chapter 10) using all kinds of computers, including Unix systems, Windows NT machines, Macintoshes, VMS VAXes, and others.

Having said this, we must acknowledge that this book is strongly oriented towards Unix (including Linux), with Windows NT as a major secondary theme. There are several reasons for this orientation. First, these operating systems are the dominant operating systems in the Internet world. Unix is still the predominant operating system for Internet servers, although Windows NT is a strong presence. Another reason is, of course, that our own experience is primarily in the Unix world; we have entered the world of Windows NT only recently, as it started to intersect with the world of the Internet. Although we do speak Windows NT, we do so with a strong Unix accent.

Linux, while it is not strictly speaking Unix, is a close relative of the Unix we have spent our careers working with. In many cases, it is truer to the Unix tradition than commercial operating systems entitled to use the Unix trademark. While we do mention Linux by name in some places, you should bear in mind that all of our statements about Unix are meant to include Linux except when we explicitly state otherwise.

Similarly, when we mention "Windows NT", unless we explicitly mention versions, we mean both Windows NT 4 and Windows 2000. Windows 2000 is a direct descendant of Windows NT 4 and behaves like it in most important respects. We call out differences where appropriate (although you should bear in mind that Windows 2000 was being released as this book went to press; both the operating system and the world's experience with it are bound to have changed by the time you read this).

Products

It's impossible to give a complete list of commercial and publicly available products in this book because new products are constantly being introduced and capabilities are constantly being added to existing products. Instead, we concentrate on discussing generic features and capabilities, and the consequences of having—or not having—particular capabilities, so that you can make your own evaluation of the products currently available to you. We do periodically mention individual

products, some commercial and some publicly available, particularly when there are striking features of well-known products. This is not intended to be an endorsement of the products we mention, or a slight to products that we omit.

Examples

Writing a book of this nature requires a large number of examples with hostnames and addresses in them. In order to avoid offending or inconveniencing people, we have attempted to use only names and addresses that are not in use. In most cases, we have used names and addresses that are reserved and cannot be publicly registered. In particular, this is why most of the example hosts in this book are in the ".example" domain (reserved for this use in RFC 2606). In a few cases where we needed large numbers of hostnames and felt that using the reserved example namespace would be confusing, we have used names that can be registered; we have attempted to use names that are not currently registered and do not seem likely to be registered. We apologize to anybody who inadvertently uses one of these names and is inconvenienced.

We also apologize to those readers who have memorized the entire reserved IP address space, and find it upsetting that many of our illustrations show reserved IP addresses in use over the Internet. This is, of course, impossible in practice, and we show it only to avoid attracting undesirable attention to addresses that can be accessed over the Internet.

Conventions Used in This Book

The following conventions are used in this book:

Italic
: Used for file and directory names and URLs, and for the first mention of new terms under discussion.

`Constant width`
: Used for code examples.

`Constant width italic`
: In some code examples, indicates an element (e.g., a filename) that you supply.

The following icon is used in this book:



Indicates a tip, suggestion, or general note.

Comments and Questions

We have tested and verified the information in this book to the best of our ability, but you may find that features have changed (or even that we have made mistakes!). Please let us know about any errors you find, as well as your suggestions for future editions, by writing to:

O'Reilly & Associates
101 Morris Street
Sebastopol, CA 95472
(800) 998-9938 (in the United States or Canada)
(707) 829-0515 (international or local)
(707) 829-0104 (fax)

There is a web page for this book, where we list any errata, plans for future editions, and additional information. You can access this page at:

http://www.oreilly.com/catalog/fire2/

To ask technical questions or comment on the book, send email to:

bookquestions@oreilly.com

For more information about our books, conferences, software, Resource Centers, and the O'Reilly Network, see our web site at:

http://www.oreilly.com

Acknowledgments for the Second Edition

As unlikely as it may seem, we still had no idea how much time and effort the second edition would take when we started working on it; what we expected to be a relatively simple effort has turned into a marathon. Even the smallest revision requires many hands, and a fully new edition requires what seems like a cast of thousands.

Thanks to those who reviewed the second edition and made helpful comments: Steve Beaty, David LeBlanc, Phil Cox, Eric Pearce, Chuck Phillips, Greg Rose, and Wietse Venema—and to Bruce Schneier and Diana Smetters who read Appendix C on a four-hour turnaround! Thanks to the entire editorial and production team at O'Reilly, especially project manager Madeleine Newell and production editor Nancy Crumpton.

Elizabeth says: My thanks to my friends, family, and colleagues for their patience and aid; my monomaniacal interest in network protocols coupled with emotional instability and intermittent overwork have required more than a reasonable and

customary amount of tolerance. I am particularly indebted to Arnold Zwicky, Diana Smetters, Jeanne Dusseault, and Brent Chapman. Special thanks are due to my second father, Jacques Transue, who required me to take slow and calm breaks from writing. Thanks to Debby Russell and Sue Miller at O'Reilly for their deft, patient, and calm job of editing; and to Simon, who expected a simple writing project, got his life disrupted for more than a year and a half, and kept working anyway, even though we insisted on spelling everything in American instead of proper English. And thanks to the many O'Reilly people who helped to produce this book.

Simon says: I would like to thank my colleagues, my friends, and my family for their understanding and support during this project. Particular thanks go to Beryl Cooper, Mel Pleasant, Landon Curt Noll, Greg Bossert, James R. Martin II, Alesia Bischoff, and Cherry Mill for their encouragement and patience. A special mention goes to my ice hockey teammates—thanks for such an active alternative to writing. Enormous thanks to Elizabeth for asking me to coauthor and for coaching me through the process. Finally, thanks to Debby, Sue, and the staff of O'Reilly for putting this book into the hands of our readers.

Acknowledgments for the First Edition

Note: We've preserved these acknowledgments for the first edition because we continue to be grateful to the people who helped us with that edition. Note, however, that several parts of the first edition (e.g., the foreword and the TCP/IP appendix) are no longer included in the book.

When we set out to write this book, we had no idea that it would consume so much time and energy. We would never have succeeded without the help of many people.

Special thanks to Ed DeHart and Craig Hunt. Ed worked with Brent in the early stages of this book and wrote the foreword to it; we appreciate all that he has done to help. TCP/IP is essential for understanding the basics of firewall construction, and Craig Hunt, author of *TCP/IP Network Administration* (O'Reilly & Associates) has kindly let us excerpt much of that book's Chapters 1 and 2 in this book's Appendix C so readers who do not already have a TCP/IP background can get a jump start.

Thanks to all those who reviewed drafts of the book before publication and made helpful suggestions: Fred Avolio, Steve Bellovin, Niels Bjergstrom, Rik Farrow, Simson Garfinkel, Eliot Lear, Evi Nemeth, Steve Simmons, Steve Romig, Gene Spaf-

ford, Phil Trubey, and Mark Verber. Thanks as well to Eric Allman for answering many Sendmail questions and Paul Traina for answering many Cisco questions.

Thanks to all the people at O'Reilly & Associates who turned this manuscript into a finished book: to Mary Anne Weeks Mayo, the wonderful and patient project manager/copyeditor for the book; Len Muellner, Ellen Siever, and Norm Walsh, who converted the book from Word to SGML and contributed their tool-tweaking prowess; Chris Reilley, who created the many excellent diagrams; Edie Freedman, who designed the cover, and Nancy Priest, who designed the interior layout; John Files and Juliette Muellner, who assisted with production; Seth Maislin, who prepared the index; and Sheryl Avruch and Kismet McDonough-Chan, who did the final quality control on the book.

Brent says: I would like to extend personal thanks to my friends and family, for keeping me going for a year and a half while I worked on the book; to my staff at Great Circle Associates, for keeping my business going; to the many hundreds of folks who've attended my Internet Security Firewalls Tutorial, for providing the impetus for this whole endeavor (and for keeping my bills paid!); and to the many thousands of subscribers to the Firewalls mailing list on the Internet, for providing a stimulating environment to develop many of the ideas found in this book. I also owe a lot of thanks to Debby Russell, our editor at O'Reilly & Associates, for all her help and guidance, and to our technical reviewers, for all their wonderful comments and suggestions. Most of all, though, I'd like to thank my very good friend and coauthor, Elizabeth Zwicky, without whose collaboration and encouragement this book probably never would have been finished, and certainly wouldn't have been as good.

Elizabeth says: My thanks go to my friends, my family, and my colleagues at Silicon Graphics, for an almost infinite patience with my tendency to alternate between obsessing about the book and refusing to discuss anything even tangentially related to it. I'd like to particularly thank Arnold Zwicky, Diana Smetters, Greg Rose, Eliot Lear, and Jeanne Dusseault for their expert moral support (often during similar crises of their own). But the most thanks for this effort have to go to Debby and Brent, for giving me a chance to be part of an unexpected but extremely rewarding project.

5

Firewall Technologies

In Part I, we introduced Internet firewalls and summarized what they can and cannot do to improve network security. In this chapter, we present major firewalls concepts. What are the terms you will hear in discussions of Internet firewalls? What are the components that can be put together to build these common firewall architectures? How do you evaluate a firewall design? In the remaining chapters of this book, we'll describe these components and architectures in detail.

Some Firewall Definitions

You may be familiar with some of the following firewall terms, and some may be new to you. Some may seem familiar, but they may be used in a way that is slightly different from what you're accustomed to (though we try to use terms that are as standard as possible). Unfortunately, there is no completely consistent terminology for firewall architectures and components. Different people use terms in different—or, worse still, conflicting—ways. Also, these same terms sometimes have other meanings in other networking fields; the following definitions are for a firewalls context.

Here are some very basic definitions; we describe these terms in greater detail elsewhere:

Firewall
: A component or set of components that restricts access between a protected network and the Internet, or between other sets of networks.

Host
: A computer system attached to a network.

Bastion host

A computer system that must be highly secured because it is vulnerable to attack, usually because it is exposed to the Internet and is a main point of contact for users of internal networks. It gets its name from the highly fortified projections on the outer walls of medieval castles.*

Dual-homed host

A general-purpose computer system that has at least two network interfaces (or homes).

Network address translation (NAT)

A procedure by which a router changes data in packets to modify the network addresses. This allows a router to conceal the addresses of network hosts on one side of it. This technique can enable a large number of hosts to connect to the Internet using a small number of allocated addresses or can allow a network that's configured with illegal or unroutable addresses to connect to the Internet using valid addresses. It is not actually a security technique, although it can provide a small amount of additional security. However, it generally runs on the same routers that make up part of the firewall.

Packet

The fundamental unit of communication on the Internet.

Packet filtering

The action a device takes to selectively control the flow of data to and from a network. Packet filters allow or block packets, usually while routing them from one network to another (most often from the Internet to an internal network, and vice versa). To accomplish packet filtering, you set up a set of rules that specify what types of packets (e.g., those to or from a particular IP address or port) are to be allowed and what types are to be blocked. Packet filtering may occur in a router, in a bridge, or on an individual host. It is sometimes known as *screening*.†

Perimeter network

A network added between a protected network and an external network, in order to provide an additional layer of security. A perimeter network is sometimes called a *DMZ*, which stands for *De-Militarized Zone* (named after the zone separating North and South Korea).

---

* Marcus Ranum, who is generally held responsible for the popularity of this term in the firewalls professional community, says, "Bastions . . . overlook critical areas of defense, usually having stronger walls, room for extra troops, and the occasional useful tub of boiling hot oil for discouraging attackers".

† Some networking literature (in particular, the BSD Unix release from Berkeley) uses the term "packet filtering" to refer to something else entirely (selecting certain packets off a network for analysis, as is done by the *etherfind* or *tcpdump* programs).

Proxy

A program that deals with external servers on behalf of internal clients. Proxy clients talk to proxy servers, which relay approved client requests on to real servers, and relay answers back to clients.

Virtual private network (VPN)

A network where packets that are internal to a private network pass across a public network, without this being obvious to hosts on the private network. In general, VPNs use encryption to protect the packets as they pass across the public network. VPN solutions are popular because it is often cheaper to connect two networks via public networks (for instance, getting them both Internet connections) than via private networks (like traditional leased-line connections between the sites).

The next few sections briefly describe the major technologies associated with firewalls: packet filtering, proxy services, network address translation, and virtual private networks.

There are legitimate questions about how to distinguish between packet filtering and proxying, particularly when dealing with complex packet filtering systems and simple proxies. Many people believe that systems that pay attention to individual protocols and/or modify packets should not be considered packet filters, and may even refer to these systems as *transparent proxies.* In fact, these systems don't behave much like older, simpler packet filtering systems, and it's a good idea not to apply generalizations about packet filtering to them blindly. On the other hand, they don't behave much like proxying systems, either.

Similarly, a number of proxying systems provide *generic proxies,* which essentially function like packet filters, accepting all traffic to a given port without analyzing it. It's advisable to pay close attention to the individual technology a product uses, without making assumptions based on whether it claims to be a packet filter or a proxy. However, many systems still are clearly packet filters or clearly proxies, so it is worth understanding what these technologies are and how they work.

Packet Filtering

Packet filtering systems route packets between internal and external hosts, but they do it selectively. They allow or block certain types of packets in a way that reflects a site's own security policy, as shown in Figure 5-1. The type of router used in a packet filtering firewall is known as a *screening router.*

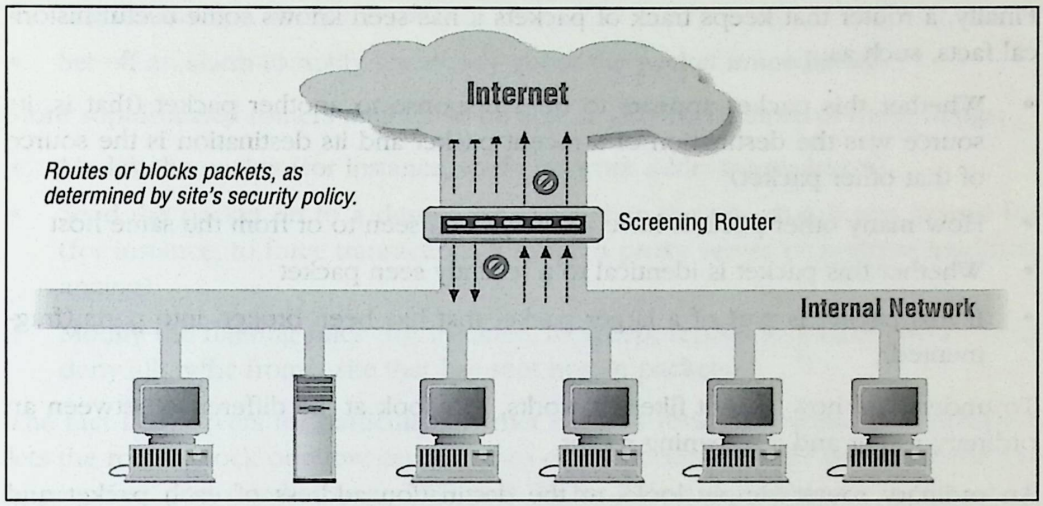


Figure 5-1. Using a screening router to do packet filtering

As we discuss in Chapter 8, *Packet Filtering*, every packet has a set of headers containing certain information. The main information is:

- IP source address
- IP destination address
- Protocol (whether the packet is a TCP, UDP, or ICMP packet)
- TCP or UDP source port
- TCP or UDP destination port
- ICMP message type
- Packet size

The router can also look past the packet headers at data further on in the packet; this allows it, for instance, to filter packets based on more detailed information (like the name of the web page that somebody is requesting) and to verify that packets appear to be formatted as expected for their destination port. The router can also make sure that the packet is valid (it actually is the size that it claims to be and is a legal size, for instance), which helps catch a number of denial of service attacks based on malformed packets.

In addition, the router knows things about the packet that aren't reflected in the packet itself, such as:

- The interface the packet arrives on
- The interface the packet will go out on

Finally, a router that keeps track of packets it has seen knows some useful historical facts, such as:

- Whether this packet appears to be a response to another packet (that is, its source was the destination of a recent packet and its destination is the source of that other packet)
- How many other packets have recently been seen to or from the same host
- Whether this packet is identical to a recently seen packet
- If this packet is part of a larger packet that has been broken into parts (fragmented)

To understand how packet filtering works, let's look at the difference between an ordinary router and a screening router.

An ordinary router simply looks at the destination address of each packet and picks the best way it knows to send that packet towards that destination. The decision about how to handle the packet is based solely on its destination. There are two possibilities: the router knows how to send the packet towards its destination, and it does so; or the router does not know how to send the packet towards its destination, and it forgets about the packet and returns an ICMP "destination unreachable" message, to the packet's source.

A screening router, on the other hand, looks at packets more closely. In addition to determining whether or not it *can* route a packet towards its destination, a screening router also determines whether or not it *should*. "Should" or "should not" are determined by the site's security policy, which the screening router has been configured to enforce.

Packet filtering may also be performed by devices that pay attention only to "should" and "should not" and have no ability to route. Devices that do this are *packet filtering bridges*. They are rarer than packet filtering routers, mostly because they are dedicated security devices that don't provide all the other functions that routers do. Most sites would rather add features to routers that they need anyway, instead of adding a dedicated device. However, being a dedicated device provides advantages for packet filtering bridges; in particular, they are harder to detect and attack than packet filtering routers. They provide the same general features that we discuss for packet filtering routers.

Once it has looked at all the information, a straightforward packet filtering router can do any of the following things:

- Send the packet on to the destination it was bound for.
- Drop the packet—just forget it, without notifying the sender.
- Reject the packet—refuse to forward it, and return an error to the sender.

- Log information about the packet.
- Set off an alarm to notify somebody about the packet immediately.

More sophisticated routers might also be able to do one or more of these things:

- Modify the packet (for instance, to do network address translation).
- Send the packet on to a destination other than the one that it was bound for (for instance, to force transactions through a proxy server or perform load balancing).
- Modify the filtering rules (for instance, to accept replies to a UDP packet or to deny all traffic from a site that has sent hostile packets).

The fact that servers for particular Internet services reside at certain port numbers lets the router block or allow certain types of connections simply by specifying the appropriate port number (e.g., TCP port 23 for Telnet connections) in the set of rules specified for packet filtering. (Chapter 8, *Packet Filtering*, describes in detail how you construct these rules.)

Here are some examples of ways in which you might program a screening router to selectively route packets to or from your site:

- Block all incoming connections from systems outside the internal network, except for incoming SMTP connections (so that you can receive electronic mail).
- Block all connections to or from certain systems you distrust.
- Allow electronic mail and FTP services, but block dangerous services like TFTP, the X Window System, RPC, and the "r" services (*rlogin*, *rsh*, *rcp*, etc.). (See Chapter 13, *Internet Services and Firewalls*, for more information.)

Packet filtering devices that keep track of packets that they see are frequently called *stateful packet filters* (because they keep information about the state of transactions). They may also be called *dynamic packet filters* because they change their handling of packets dynamically depending on the traffic they see. Devices that look at the content of packets, rather than at just their headers, are frequently called *intelligent packet filters*. In practice, almost all stateful packet filters also are capable of looking at the contents of packets, and many are also capable of modifying the contents of packets, so you may see all these capabilities lumped together under the heading "stateful packet filtering". However, something can legitimately be called a "stateful packet filter" without having the ability to do advanced content filtering or modification.

A packet filtering system is also a logical place to provide virtual private network or network address translation services. Since the packet filter is already looking at all of the packets, it can easily identify packets that are intended for a destination

that is part of the virtual private network, encrypt those packets, and encapsulate them in another packet bound for the appropriate destination.

Advantages of Packet Filtering

Packet filtering has a number of advantages.

One screening router can help protect an entire network

One of the key advantages of packet filtering is that a single, strategically placed packet filtering router can help protect an entire network. If only one router connects your site to the Internet, you gain tremendous leverage on network security, regardless of the size of your site, by doing packet filtering on that router.

Simple packet filtering is extremely efficient

Because simple packet filtering requires paying attention only to a few packet headers, it can be done with very low overhead. Proxying is a fairly time-consuming operation, and adding proxying means directing connections through another program, usually on a machine that otherwise wouldn't be necessary to the routing process. Packet filtering takes place on a machine that was already in the critical path, and introduces a much smaller delay.

However, there is no free lunch; the more work your packet filters do, the slower they will be. If your packet filters behave like proxies, doing complicated data-driven operations that require keeping track of multiple packets, they will tend to perform like proxies as well.

Packet filtering is widely available

Packet filtering capabilities are available in many hardware and software routing products, both commercial and freely available over the Internet. Most sites already have packet filtering capabilities available in the routers they use.

Most commercial router products include packet filtering capabilities. Packet filtering capabilities are also available for a number of general-purpose computers. These are discussed further in Chapter 8, *Packet Filtering*.

Disadvantages of Packet Filtering

Although packet filtering provides many advantages, there are some disadvantages to using packet filtering as well.

Current filtering tools are not perfect

Despite the widespread availability of packet filtering in various hardware and software packages, packet filtering is still not a perfect tool. The packet filtering capabilities of many of these products share, to a greater or lesser degree, common limitations:

- The packet filtering rules tend to be hard to configure. Although there is a range of difficulty, it mostly runs from slightly mind-twisting to brain-numbingly impossible.
- Once configured, the packet filtering rules tend to be hard to test.
- The packet filtering capabilities of many of the products are incomplete, making implementation of certain types of highly desirable filters difficult or impossible.
- Like anything else, packet filtering packages may have bugs in them; these bugs are more likely than proxying bugs to result in security problems. Usually, a proxy that fails simply stops passing data, while a failed packet filtering implementation may allow packets it should have denied.

Packet filtering reduces router performance

Doing packet filtering places a significant extra load on a router. As we discussed previously, more complex filters place more load on the router, but in some cases, simply turning on packet filtering on a given interface can also cost you a lot of performance on some routers, because the filtering is incompatible with certain caching strategies commonly used for performance enhancement. Cisco's "fastpath" functionality is an example of this; normally, fastpath can perform basic routing functions completely on the interface card, without involving the main CPU, but using some forms of filtering requires involving the main CPU for each packet, which is much slower. What enables/disables fastpath depends on the hardware and software version.

Some policies can't readily be enforced by normal packet filtering routers

The information that a packet filtering router has available to it doesn't allow you to specify some rules you might like to have. For example, packets say what host they come from but generally not what user. Therefore, you can't enforce restrictions on particular users. Similarly, packets say what port they're going to but not what application; when you enforce restrictions on higher-level protocols, you do it by port number, hoping that nothing else is running on the port assigned to that protocol. Malicious insiders can easily subvert this kind of control.

This problem is eased by using more intelligent packet filters; however, in each case, you have to give up some of the advantages of normal packet filtering. For

instance, a packet filter can insist that users authenticate themselves before sending packets, and then it can filter packets by username. However, this removes the transparency advantage of normal packet filtering. A packet filter can also do protocol validity checking, but this is less than perfect and also increases filtering overhead.

Proxy Services

In general, a *proxy* is something or someone who does something on somebody else's behalf. For instance, you may give somebody the ability to vote for you by proxy in an election.

Proxy services are specialized application or server programs that take users' requests for Internet services (such as FTP and Telnet) and forward them to the actual services. The proxies provide replacement connections and act as gateways to the services. For this reason, proxies are sometimes known as *application-level gateways.*• In this book, when we are talking about proxy services, we are specifically talking about proxies run for security purposes, which are run on a firewall host: either a dual-homed host with an interface on the internal network and one on the external network, or some other bastion host that has access to the Internet and is accessible from the internal machines.

You will also run into proxies that are primarily designed for network efficiency instead of for security; these are *caching proxies,* which keep copies of the information for each request that they proxy. The advantage of a caching proxy is that if multiple internal hosts request the same data, the data can be provided directly by the proxy. Caching proxies can significantly reduce the load on network connections. There are proxy servers that provide both security and caching; in general, they are better at one purpose than the other.

Proxy services sit, more or less transparently, between a user on the inside (on the internal network) and a service on the outside (on the Internet). Instead of talking to each other directly, each talks to a proxy. Proxies handle all the communication between users and Internet services behind the scenes.

Transparency is the major benefit of proxy services. It's essentially smoke and mirrors. To the user, a proxy server presents the illusion that the user is dealing directly with the real server. To the real server, the proxy server presents the illu-

---

• Firewall terminologies differ. Whereas we use the term *proxy service* to encompass the entire proxy approach, other authors refer to *application-level gateways* and *circuit-level gateways.* Although there are small differences between the meanings of these various terms, which we'll explore in Chapter 9, *Proxy Systems,* in general our discussion of proxies refers to the same type of technology other authors mean when they refer to these gateway systems.

sion that the real server is dealing directly with a user on the proxy host (as opposed to the user's real host).

How do proxy services work? Let's look at the simplest case, where we add proxy services to a dual-homed host. (We'll describe these hosts in some detail in the "Dual-Homed Host" section in Chapter 6, *Firewall Architectures.*)



Proxy services are effective only when they're used in conjunction with a mechanism that restricts direct communications between the internal and external hosts. Dual-homed hosts and packet filtering are two such mechanisms. If internal hosts are able to communicate directly with external hosts, there's no need for users to use proxy services, and so (in general) they won't. Such a bypass probably isn't in accordance with your security policy.

As Figure 5-2 shows, a proxy service requires two components: a proxy server and a proxy client. In this illustration, the *proxy server* runs on the dual-homed host (as we discuss in Chapter 9, *Proxy Systems*, there are other ways to set up a proxy server). A *proxy client* is a special version of a normal client program (e.g., a Telnet or FTP client) that talks to the proxy server rather than to the "real" server out on the Internet; in some configurations, normal client programs can be used as proxy clients. The proxy server evaluates requests from the proxy client and decides which to approve and which to deny. If a request is approved, the proxy server contacts the real server on behalf of the client (thus the term *proxy*) and proceeds to relay requests from the proxy client to the real server, and responses from the real server to the proxy client.

In some proxy systems, instead of installing custom client proxy software, you'll use standard software but set up custom user procedures for using it. (We'll describe how this works in Chapter 9.)

There are also systems that provide a hybrid between packet filtering and proxying where a network device intercepts the connection and acts as a proxy or redirects the connection to a proxy; this allows proxying without making changes to the clients or the user procedures.

The proxy server doesn't always just forward users' requests on to the real Internet services. The proxy server can control what users do because it can make decisions about the requests it processes. Depending on your site's security policy, requests might be allowed or refused. For example, the FTP proxy might refuse to let users export files, or it might allow users to import files only from certain sites. More sophisticated proxy services might allow different capabilities to different hosts, rather than enforcing the same restrictions on all hosts.

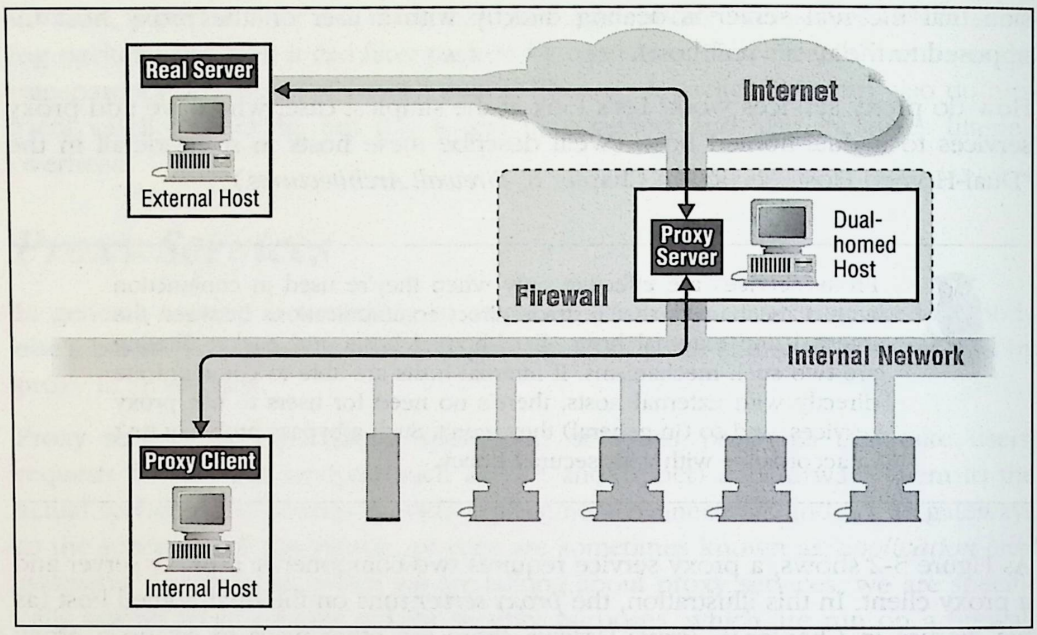


Figure 5-2. Using proxy services with a dual-homed host

Some proxy servers do in fact just forward requests on, no matter what they are. These may be called *generic proxies* or *port forwarders*. Programs that do this are providing basically the same protections that you would get if you had a packet filter in place that was allowing traffic on that port. You do not get any significant increase in security by replacing packet filters with proxies that do exactly the same thing (you gain some protection against malformed packets, but you lose by adding an attackable proxying program).

Some excellent software is available for proxying. SOCKS is a proxy construction toolkit, designed to make it easy to convert existing client/server applications into proxy versions of those same applications. The Trusted Information Systems Internet Firewall Toolkit (TIS FWTK) includes proxy servers for a number of common Internet protocols, including Telnet, FTP, HTTP, *rlogin*, X11, and others; these proxy servers are designed to be used in conjunction with custom user procedures. See the discussion of these packages in Chapter 9, *Proxy Systems*.

Many standard client and server programs, both commercial and freely available, now come equipped with their own proxying capabilities or with support for generic proxy systems like SOCKS. These capabilities can be enabled at runtime or compile time.

Most proxy systems are used to control and optimize outbound connections; they are controlled by the site where the clients are. It is also possible to use proxy systems to control and optimize inbound connections to servers (for instance, to

balance connections among multiple servers or to apply extra security). This is sometimes called *reverse proxying*.

Advantages of Proxying

There are a number of advantages to using proxy services.

Proxy services can be good at logging

Because proxy servers can understand the application protocol, they can allow logging to be performed in a particularly effective way. For example, instead of logging all of the data transferred, an FTP proxy server can log only the commands issued and the server responses received; this results in a much smaller and more useful log.

Proxy services can provide caching

Since all requests are passing through the proxy service anyway, the proxy can provide caching, keeping local copies of the requested data. If the number of repeat requests is significant, caching can significantly increase performance and reduce the load on network links.

Proxy services can do intelligent filtering

Since a proxy service is looking at specific connections, it is frequently able to do filtering more intelligently than a packet filter. For instance, proxy services are much more capable of filtering HTTP by content type (for instance, to remove Java or JavaScript) and better at virus detection than packet filtering systems.

Proxy systems can perform user-level authentication

Because a proxy system is actively involved in the connection, it is easy for it to do user authentication and to take actions that depend on the user involved. Although this is possible with packet filtering systems, it is much more difficult.

Proxy systems automatically provide protection for weak or faulty IP implementations

As a proxy system sits between a client and the Internet, it generates completely new IP packets for the client. It can therefore protect clients from deliberately malformed IP packets. (You just need a proxy system that isn't vulnerable to the bad packets!)

Disadvantages of Proxying

There are also some disadvantages to using proxy services.

Proxy services lag behind nonproxied services

Although proxy software is widely available for the older and simpler services like FTP and Telnet, proven software for newer or less widely used services is harder to find. There's usually a distinct lag between the introduction of a service and the availability of proxying servers for it; the length of the lag depends primarily on how well the service is designed for proxying. This makes it difficult for a site to offer new services immediately as they become available. Until suitable proxy software is available, a system that needs new services may have to be placed outside the firewall, opening up potential security holes. (Some services can be run through generic proxies, which will give at least minimal protection.)

Proxy services may require different servers for each service

You may need a different proxy server for each protocol, because the proxy server may need to understand the protocol in order to determine what to allow and disallow, and in order to masquerade as a client to the real server and as the real server to the proxy client. Collecting, installing, and configuring all these various servers can be a lot of work. Again, you may be able to use a generic proxy, but generic proxies provide only the same sorts of protection and functionality that you could get from packet filters.

Products and packages differ greatly in the ease with which they can be configured, but making things easier in one place can make it harder in others. For example, servers that are particularly easy to configure can be limited in flexibility; they're easy to configure because they make certain assumptions about how they're going to be used, which may or may not be correct or appropriate for your site.

Proxy services usually require modifications to clients, applications, or procedures

Except for services designed for proxying, you will need to use modified clients, applications, and/or procedures. These modifications can have drawbacks; people can't always use the readily available tools with their normal instructions.

Because of these modifications, proxied applications don't always work as well as nonproxied applications. They tend to bend protocol specifications, and some clients and servers are less flexible than others.

Network Address Translation

Network address translation (NAT) allows a network to use one set of network addresses internally and a different set when dealing with external networks. Network address translation does not, by itself, provide any security, but it helps to

conceal the internal network layout and to force connections to go through a choke point (because connections to untranslated addresses will not work, and the choke point does the translation).

Like packet filtering, network address translation works by having a router do extra work. In this case, not only does the router send packets on, but it also modifies them. When an internal machine sends a packet to the outside, the network address translation system modifies the source address of the packet to make the packet look as if it is coming from a valid address. When an external machine sends a packet to the inside, the network address translation system modifies the destination address to turn the externally visible address into the correct internal address. The network address translation system can also modify the source and destination port numbers (this is sometimes called Port and Address Translation or PAT). Figure 5-3 shows a network address translation system modifying only addresses, while Figure 5-4 shows port and address translation.

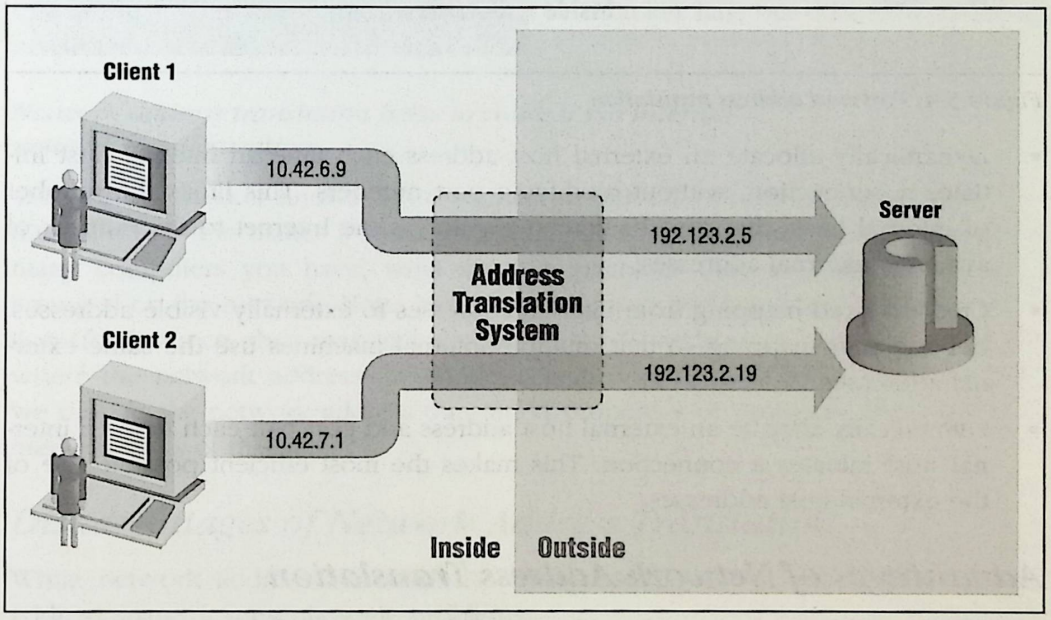


Figure 5-3. Network address translation

Network address translation systems can use different schemes for translating between internal and external addresses:

- Allocate one external host address for each internal address and always apply the same translation. This provides no savings in address space, and it slows down connections; it is normally a temporary measure used by sites that have been using illegal address spaces but are in the process of moving to using valid addresses.

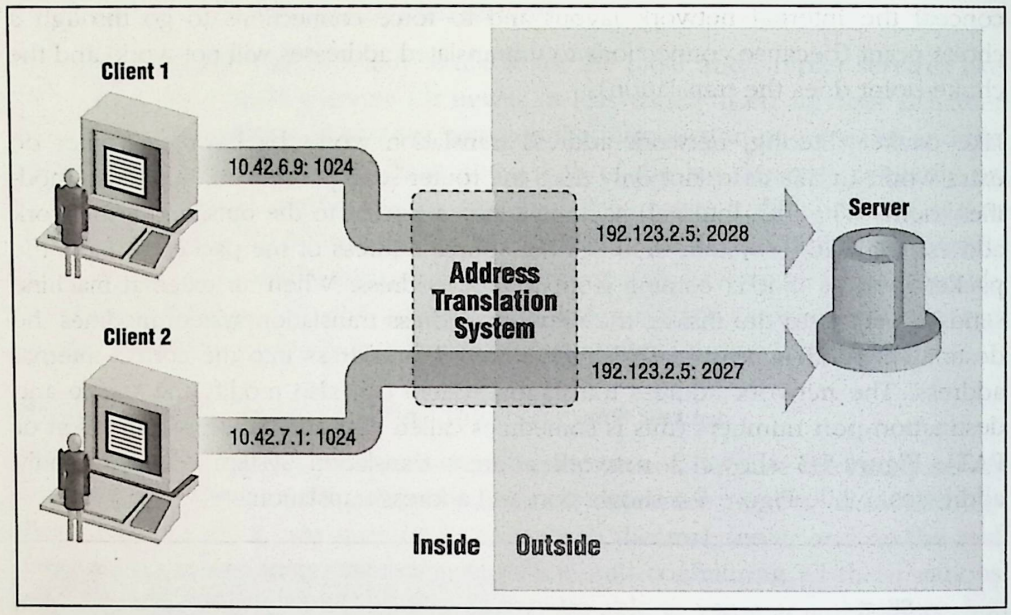


Figure 5-4. Port and address translation

- Dynamically allocate an external host address each time an internal host initiates a connection, without modifying port numbers. This limits the number of internal hosts that can simultaneously access the Internet to the number of available external addresses.
- Create a fixed mapping from internal addresses to externally visible addresses, but use port mapping so that multiple internal machines use the same external addresses.
- Dynamically allocate an external host address and port pair each time an internal host initiates a connection. This makes the most efficient possible use of the external host addresses.

Advantages of Network Address Translation

The main purpose of network address translation is to economize on address space, but it can also have some security advantages.

Network address translation helps to enforce the firewall's control over outbound connections

Since individual hosts have addresses that won't work on the external network, they require the assistance of the network address translation system to connect. If a host finds a way to connect to the Internet without going through the address translation, the connection won't work.

Network address translation can help restrict incoming traffic

Depending on how you configure a network address translation system, it can provide stronger restrictions on incoming traffic than packet filtering. A network address translation system that's doing dynamic translation will allow only packets that are part of a current interaction initiated from the inside. This is similar to the protection that a dynamic packet filter offers, but the changing IP addresses put stronger time constraints on attackers. Not only can they attack only certain ports, but if they wait too long, the address translation will have gone away, and the entire address will have disappeared or been given to another host.

Many people assume that all network address translation systems provide this sort of protection, but this is not true. If you configure a network address translation system to do static translations, it may provide no restrictions at all on incoming traffic. Even doing dynamic translations, the simplest implementations allocate an entire externally visible address to the internal host and translate all traffic sent to that address. This does limit the time that an attacker has, but otherwise provides no protection at all.

Network address translation helps to conceal the internal network's configuration

The less an attacker knows about you, the better off you are. A network address translation system makes it much more difficult for an attacker to determine how many computers you have, what kind of machines they are, and how they're arranged on the network. Note, however, that many protocols leak useful information (for instance, they may include the client's IP address or hostname in places where the network address translation system doesn't need to change it). When we discuss the network address translation properties of protocols, we attempt to mention leaks of this sort.

Disadvantages of Network Address Translation

While network address translation is a very useful way of conserving network address space, it presents some problems.

Dynamic allocation requires state information that is not always available

It is very easy for a network address translation system to tell whether or not a host has stopped using a TCP connection, but there's no way to know at the packet header level whether a UDP packet is part of an ongoing conversation or is an isolated event. This means that a network address translation system has to guess how long it should keep a particular translation. If it guesses incorrectly, responses may be lost or delivered to unexpected hosts.

Embedded IP addresses are a problem for network address translation

Network address translation systems normally translate the addresses in the headers of packets (see Chapter 4, *Packets and Protocols*, for more information about packet layout). Some protocols also hide addresses in other places, and in order to find those addresses, the network address translator has to understand the protocol enough to find and modify the address, while preserving the validity of the packet. Most network address translation systems are capable of doing this for at least some protocols (for instance, FTP) but not for all protocols.

Network address translation interferes with some encryption and authentication systems

Systems for encrypting data often attempt to ensure the integrity of the data, so that the systems that are communicating know that packets have not been tampered with in transit. Network address translation is a form of tampering with the data in transit. If the protocol that's being translated does not protect the data that the network address translation system modifies, it will work. Otherwise, the integrity checking will be violated, and connections will fail. In most cases, protocols that do not have embedded IP addresses are compatible (the packet headers are not part of the protocol's protected data). The major exception to this rule is IPsec, which protects the entire packet, including headers. Network address translation is almost guaranteed to fail for protocols that combine embedded IP addresses with data integrity protection.

Dynamic allocation of addresses interferes with logging

If you are logging information after the network address translation happens, the logs will show the translated addresses, and you will have to correlate the logs with information from the network address translation system to figure out what internal system is actually involved. For instance, if you have a screened subnet architecture (discussed in Chapter 6, *Firewall Architectures*), and you are doing network address translation on the interior router, the translated addresses will be in logs from the exterior router or from a caching web proxy server on the screened subnet. Although log correlation is theoretically possible, it may be difficult, and clock synchronization will be critical.

Dynamic allocation of ports may interfere with packet filtering

Packet filtering systems pay attention to source and destination port numbers in order to try to figure out what protocol a packet should be using. Changing the source port may change the packet's acceptability. In most cases, this is not a problem because address translation systems are translating for clients, which are usually allowed to use any port above 1023. However, if ports above 1023 are translated to ports below 1023, traffic may be dropped.

Virtual Private Networks

A *virtual private network* (VPN) is a way of employing encryption and integrity protection so that you can use a public network (for instance, the Internet) as if it were a private network (a piece of cabling that you control). Making a private, high-speed, long-distance connection between two sites is much more expensive than connecting the same two sites to a public high-speed network, but it's also much more secure. A virtual private network is an attempt to combine the advantages of a public network (it's cheap and widely available) with some of the advantages of a private network (it's secure).

Fundamentally, all virtual private networks that run over the Internet employ the same principle: traffic is encrypted, integrity protected, and encapsulated into new packets, which are sent across the Internet to something that undoes the encapsulation, checks the integrity, and decrypts the traffic.

Virtual private networks are not exactly a firewall technology, but we discuss them here for several reasons:

- If you're using virtual private networking, you need to be careful about how it interacts with the firewall. In many cases, the firewall can't control traffic that comes in over the virtual network, which makes it a way to avoid the firewall controls and open new insecurities.
- A firewall is a convenient place to add virtual private networking features.
- We will frequently mention virtual private networking as a way to provide remote services that cannot be provided securely using other firewall techniques.

Where Do You Encrypt?

Virtual private networks depend on encryption. That encryption can be done as a transport method, where a host decides to encrypt traffic when it is generated, or as a tunnel, where traffic is encrypted and decrypted somewhere in between the source and the destination. The question of where you do the encryption and decryption relative to your packet filtering is an important one. If you do the encryption and decryption inside the packet filtering perimeter (i.e., on your internal net), then the filters just have to allow the encrypted packets in and out. This is especially easy if you're doing tunneling, because all the tunneled packets will be addressed to the same remote address and port number at the other end of the tunnel (the decryption unit). On the other hand, doing the encryption and decryption inside your filtering perimeter means that packets arriving encrypted are not subject to the scrutiny of the packet filters. This leaves you vulnerable to attack from the other site if that site has been compromised.

If you do the encryption and decryption outside the packet filtering perimeter (i.e., on your perimeter net or in your exterior router), then the packets coming in from the other site can be subjected to the full scrutiny of your packet filtering system. On the other hand, they can also be subjected to the full scrutiny of anyone who can read traffic on your perimeter net, including intruders.

Key Distribution and Certificates

As with any encryption and integrity protection system, key distribution can be a very sticky problem. A number of choices are available, including sharing keys or using a public key system; see Appendix C, *Cryptography*, for descriptions of these systems and the advantages and disadvantages of each.

Advantages of Virtual Private Networks

Most of the advantages of virtual private networks are economic; it's cheaper to use shared public networks than it is to set up dedicated connections, whether those are leased lines between sites or modem pools that allow individual machines to connect to a central site. On the other hand, virtual private networks also provide some security advantages.

Virtual private networks provide overall encryption

A virtual private network conceals all the traffic that goes over it. Not only does it guarantee that all the information is encrypted, but it also keeps people from knowing which internal machines are being used and with what protocols. You can protect information from snooping by using individual encrypted protocols, but attackers will still have some idea what machines are talking and what kind of information they're exchanging (for instance, if you use an encrypted mail protocol, they will know that things are being mailed). A virtual private network conceals more information.

Virtual private networks allow you to remotely use protocols that are difficult to secure any other way

Some protocols are extremely difficult to provide securely through a firewall. For instance, a number of protocols used on Microsoft systems are based on SMB, which provides a wide variety of services with different security implications over the same ports and connections. Packet filtering and proxying both have trouble adding security to SMB. Virtual private networking provides a way to give remote access for these protocols without letting people attack them from the Internet at large.

Disadvantages of Virtual Private Networks

Although virtual private networks are an important security tool, they also present problems in a firewall environment.

Virtual private networks involve dangerous network connections

A virtual private network runs over an actual network, which is presumably not a private network. The hosts on the virtual private network must be connected to that actual network, and if you're not careful, they will be vulnerable to attack from that network. For instance, if you use a virtual private network to provide connectivity to your internal network for mobile users who connect to the Internet, their machines may be attacked from the Internet.

Ideally, a virtual private network system will disable all other uses of the network interface. It's important to choose a system that will allow you to force this on the remote system. It's not good enough to have a system where the remote system is able to turn off other uses because the user on the remote system may turn networking back on. It's very tempting as a way to get rapid access to Internet resources.

Virtual private networks extend the network you must protect

When you attach something via a virtual private network, you are making it part of your internal network. If a machine on the virtual private network is broken into, the attacker will then be able to use the virtual private network to attack the rest of your site, from something that's treated as if it were inside of your local network. Virtual private networking is commonly used to give access to machines that are much more vulnerable than those that are physically on the network—for instance, laptops that are carried around in public, home machines that curious children have physical access to, and machines owned by other sites with interests and policies that are not identical to yours.

Even if the virtual private network disables other uses of the network interface it is running over, the machine may have other network interfaces. This can make it into a gateway between your network and others, inside your network's security perimeter.

Because of this, you want to be careful how you attach the virtual private network to your real private network, and how you secure the remote end. It may not be appropriate to make the virtual private network a seamless part of your internal network. Consider putting in a subsidiary firewall or at least special intrusion detection to watch for problems.

O'REILLY®

Building Internet Firewalls



In the years since the publication of the first edition of this classic reference, Internet use has exploded and e-commerce has become a daily part of business and personal life. As Internet usage has grown, so have the security threats—from password sniffers, to IP forgeries, to defacing web sites, to attacks that shut down popular web sites using distributed denial of service.

Firewalls are a very effective way to protect your system from many Internet security threats, and they're a critical component of today's computer networks. Like the first edition of *Building Internet Firewalls*, this second edition is a practical and detailed guide to designing and building firewalls and to configuring Internet services to work with firewalls. This much expanded edition covers Linux and Windows NT, as well as Unix. It describes a variety of firewall technologies (packet filtering, proxying, network address translation, virtual private networks) and architectures, and more than one hundred Internet services from email and file transfer, to web services and scripting languages, to naming, authentication, and database protocols.

The book also contains chapters on security policies, cryptography, maintaining firewalls, and responding to security incidents, as well as summaries of helpful firewall tools and other resources.

"In 1991, I described everything you (then) needed to know about firewalls in 11 pages of Practical Unix Security. *Times have definitely changed . . . and there is a lot more to know about firewalls. This book tells it all, in encyclopedic detail, and explains step-by-step where and how to apply that information. If you're computing on a network, you can't afford to be without this book."*

—Professor Gene Spafford
Purdue University CERIAS (Center for Education and Research in Information Assurance and Security)

"It is always easier to design a secure network properly than it is to correct deficiencies once it has been implemented. Building Internet Firewalls *lays out the fundamental principles and the detailed information you need to design a network correctly. This book is an essential reference for anyone involved in operational security."*

—David LeBlanc
Senior Technologist, ITG Information Security, Microsoft Corporation

"So much more than a book about firewalls! There's great information about management, threats, security in general, design for defense, policy, detection, reaction, and more—as well as all the detail needed to effectively implement and maintain a best-practice firewall."

—Greg Rose
Principal Engineer, QUALCOMM Incorporated

Visit O'Reilly on the Web at *www.oreilly.com*

ISBN 1-56592-871-7
US $49.95
CAN $74.95
9 781565 928718 90000
6 36920 92871 3

DBX33623