SYNGRESS®

1 YEAR UPGRADE
BUYER PROTECTION PLAN

Cisco® Security Specialist's Guide to PIX® Firewalls

Your Complete Guide to Cisco's PIX Firewalls

- Includes Coverage of the Cisco Secure PIX Firewall Advanced Exam (9E0-111)
- Full Coverage of the Latest PIX Firewall Operating System Version 6.2
- Complete Coverage of Configuring TurboACLs

Foreword by Ralph Troupe, President and CEO, Callisma

callisma

Vitaly Osipov
Mike Sweeney
Woody Weaver
Charles E. Riley Technical Reviewer
Umer Khan Technical Editor

solutions@syngress.com

With more than 1,500,000 copies of our MCSE, MCSD, CompTIA, and Cisco study guides in print, we continue to look for ways we can better serve the information needs of our readers. One way we do that is by listening.

Readers like yourself have been telling us they want an Internet-based service that would extend and enhance the value of our books. Based on reader feedback and our own strategic plan, we have created a Web site that we hope will exceed your expectations.

Solutions@syngress.com is an interactive treasure trove of useful information focusing on our book topics and related technologies. The site offers the following features:

- One-year warranty against content obsolescence due to vendor product upgrades. You can access online updates for any affected chapters.
- "Ask the Author" customer query forms that enable you to post questions to our authors and editors.
- Exclusive monthly mailings in which our experts provide answers to reader queries and clear explanations of complex material.
- Regularly updated links to sites specially selected by our editors for readers desiring additional reliable information on key topics.

Best of all, the book you're now holding is your key to this amazing site. Just go to **www.syngress.com/solutions**, and keep this book handy when you register to verify your purchase.

Thank you for giving us the opportunity to serve your needs. And be sure to let us know if there's anything else we can do to help you get the maximum value from your investment. We're listening.

www.syngress.com/solutions

SYNGRESS®

SYNGRESS®

1 YEAR UPGRADE
BUYER PROTECTION PLAN

Cisco® Security Specialist's Guide to PIX® Firewall

Foreword by Ralph Troupe, President and CEO, Callisma

Vitaly Osipov
Mike Sweeney
Woody Weaver
Charles E. Riley Technical Reviewer
Umer Khan Technical Editor

callisma

KEY	SERIAL NUMBER
001	27GYW9HV43
002	Q26UUN7TJM
003	STX3AD4HF5
004	Z6KB6Y2B7Y
005	T5RZU8MPD6
006	AQ8NC4E8S6
007	PH7PQ2A7EK
008	9RD7BK43HG
009	SX7V6CVPFH
010	5M39ZBVBR2

PUBLISHED BY
Syngress Publishing, Inc.
800 Hingham Street
Rockland, MA 02370

Cisco Security Specialist's Guide to PIX Firewall

Printed in the United States of America

1 2 3 4 5 6 7 8 9 0

ISBN: 1-931836-63-9

Technical Editor: Umer Khan
Technical Reviewer: Charles E. Riley
Acquisitions Editor: Catherine B. Nolan
Developmental Editor: Jonathan Babcock
Cover Designer: Michael Kavish
Page Layout and Art by: Personal Editions
Copy Editor: Darlene Bordwell
Indexer: Brenda Miller

Distributed by Publishers Group West in the United States and Jaguar Book Group in Canada.

Acknowledgments

We would like to acknowledge the following people for their kindness and support in making this book possible.

Ralph Troupe, Rhonda St. John, Emlyn Rhodes, and the team at Callisma for their invaluable insight into the challenges of designing, deploying and supporting world-class enterprise networks.

Karen Cross, Lance Tilford, Meaghan Cunningham, Kim Wylie, Harry Kirchner, Kevin Votel, Kent Anderson, Frida Yara, Jon Mayes, John Mesjak, Peg O'Donnell, Sandra Patterson, Betty Redmond, Roy Remer, Ron Shapiro, Patricia Kelly, Andrea Tetrick, Jennifer Pascal, Doug Reil, David Dahl, Janis Carpenter, and Susan Fryer of Publishers Group West for sharing their incredible marketing experience and expertise.

Duncan Enright, AnnHelen Lindeholm, David Burton, Febea Marinetti, and Rosie Moss of Elsevier Science for making certain that our vision remains worldwide in scope.

David Buckland, Wendi Wong, Daniel Loh, Marie Chieng, Lucy Chong, Leslie Lim, Audrey Gan, and Joseph Chan of Transquest Publishers for the enthusiasm with which they receive our books.

Kwon Sung June at Acorn Publishing for his support.

Jackie Gross, Gayle Voycey, Alexia Penny, Anik Robitaille, Craig Siddall, Darlene Morrow, Iolanda Miller, Jane Mackay, and Marie Skelly at Jackie Gross & Associates for all their help and enthusiasm representing our product in Canada.

Lois Fraser, Connie McMenemy, Shannon Russell, and the rest of the great folks at Jaguar Book Group for their help with distribution of Syngress books in Canada.

David Scott, Annette Scott, Geoff Ebbs, Hedley Partis, Bec Lowe, and Tricia Herbert of Woodslane for distributing our books throughout Australia, New Zealand, Papua New Guinea, Fiji Tonga, Solomon Islands, and the Cook Islands.

Winston Lim of Global Publishing for his help and support with distribution of Syngress books in the Philippines.

Contributors

C. Tate Baumrucker (CISSP, CCNP, Sun Enterprise Engineer, MCSE) is a Senior Consultant with Callisma, where he is responsible for leading engineering teams in the design and implementation of secure and highly available systems infrastructures and networks. Tate is an industry recognized subject matter expert in security and LAN/WAN support systems such as HTTP, SMTP, DNS, and DHCP. Tate has spent eight years providing technical consulting services for the Department of Defense, and other enterprise and service provider industries for companies including: American Home Products, Blue Cross and Blue Shield of Alabama, Amtrak, Iridium, National Geographic, Geico, GTSI, Adelphia Communications, Digex, Cambrian Communications, and BroadBand Office. Tate has also contributed to the book *Managing Cisco Network Security, Second Edition* (Syngress Publishing, ISBN: 1-931836-56-6).

Brian Browne (CISSP) is a Senior Consultant with Callisma. He provides senior-level strategic and technical security consulting to Callisma clients, has 12 years of experience in the field of information systems security, and is skilled in all phases of the security lifecycle. A former independent consultant, Brian has provided security consulting for multiple Fortune 500 clients, has been published in *Business Communications Review*, and was also a contributor to the book *Managing Cisco Network Security, Second Edition* (Syngress Publishing, ISBN: 1-931836-56-6). His security experience includes network security, firewall architectures, virtual private networks (VPNs), intrusion detection systems (IDSs), UNIX security, Windows NT security, and public key infrastructure (PKI). Brian resides in Willow Grove, PA with his wife, Lisa, and daughter, Marisa.

Vitaly Osipov (CISSP, CCSE, CCNA) is co-author for Syngress Publishing's *Check Point Next Generation Security Administration* (ISBN: 1-928994-74-1) and *Managing Cisco Network Security, Second Edition* (ISBN: 1-931836-56-6). Vitaly has spent the last six years working as a consultant for companies in Eastern, Central, and Western Europe. His

specialty is designing and implementing information security solutions. Currently Vitaly is the team leader for the consulting department of a large information security company. In his spare time, he also lends his consulting skills to the anti-spam company, CruelMail.com. Vitaly would like to extend his thanks to his many friends in the British Isles, especially the one he left in Ireland.

Derek Schatz (CISSP) is a Senior Consultant with Callisma, and is the lead Callisma resource for security in the western region of the United States. He specializes in information security strategy and the alignment of security efforts with business objectives. Derek has a broad technical background; previous positions have included stints with a Big Five consulting firm, where he managed a team in the technology risk consulting practice, and as a Systems Engineer at Applied Materials, where he was responsible for their Internet and Extranet infrastructure. Derek holds a bachelor's degree from the University of California, Irvine, and is a member of the Information Systems Security Association. He received his CISSP certification in 1999. Derek resides in Southern California with his family.

Timothy "TJ" Schuler (CCIE #8800) works as a Senior Network Engineer for Coleman Technologies in Denver, CO. TJ has over seven years of experience with network implementation and design including security, large routing and switching networks, ATM, wireless, IP Telephony and IP based video technologies. TJ is currently pursuing the Security CCIE certification, which would be his second CCIE. He would like to dedicate this work to his family.

Michael Sweeney (CCNA, CCDA, CCNP, MCSE) is the owner of the IT consulting firm, Packetattack.com. His specialties are network design, network troubleshooting, wireless network design, security, network analysis using Sniffer Pro, and wireless network analysis using AirMagnet. Michael is a graduate of the extension program at the University of California, Irvine with a certificate in Communications and Network Engineering. Michael currently resides in Orange, CA with his wife, Jeanne, and daughter, Amanda.

Robert "Woody" Weaver (CISSP) is the Field Practice Lead for Security at Callisma. As an information systems security professional, Woody's responsibilities include field delivery and professional services product development. Woody's background includes a decade as a tenured professor, teaching mathematics and computer science. Woody also spent time as the most senior Network Engineer for Williams Communications in the San Jose/San Francisco Bay area, providing client services for their network integration arm, and as Vice President of Technology for Fullspeed Network Services, a regional systems integrator. He is also a contributiong author to *Managing Cisco Network Security, Second Edition* (Syngress Publishing, ISBN: 1-931836-56-6). Woody holds a bachelor's of Science degree from the California Institute of Technology, and a Ph.D. from Ohio State. He currently works out of the Washington, D.C. metro area.

Technical Reviewer and Contributor

Charles Riley (CCNP, CSS1, CISSP, CCSA, MCSE, CNE-3) is a Network Engineer with a long tenure in the networking security field. Charles has co-authored several books including *Configuring Cisco Voice Over IP, Second Edition* (Syngress Publishing ISBN: 1-931836-64-7). He has designed and implemented robust networking solutions for large Fortune 500 and privately held companies. He started with the U.S. Army at Fort Huachuca, AZ, eventually finishing his Army stretch as the Network Manager of the Seventh Army Training Command in Grafenwoehr, Germany. Currently Charles is employed as a Network Security Engineer for HyperVine (www.hypervine.net) in Kansas, where he audits and hardens the existing security of customers, as well as deploying new security architectures and solutions. Charles holds a bachelor's degree from the University of Central Florida. He is grateful to his wife, René, and daughter, Tess, for their support of his writing: My world is better with y ou in it.

Technical Editor and Contributor

Umer Khan (CCIE #7410, MCSE, SCSA, SCNA, CCA, SCE, CNX) is the Manager of Networking and Security at Broadcom Corporation (www.broadcom.com). Umer's department is responsible for the design and implementation of global LAN/MAN/WAN solutions that are available with 99.9% up time (planned and unplanned), as well as all aspects of information security. Among other technologies, Broadcom's network consists of Cisco switching gear end-to-end, dark fiber, OC-48 SONET, DWDM, 802.11 wireless, multi-vendor virtual private networks (VPNs), and voice over IP (VoIP) technology. The information security group deals with policies, intrusion detection and response, strong authentication, and firewalls. Umer has contributed to several other books, including the *Sun Certified System Administrator for Solaris 8 Study Guide* (ISBN: 007-212369-9) and *Sniffer Pro Network Optimization & Troubleshooting Handbook* (Syngress Publishing, ISBN: 1-931836-57-4). Umer received a bachelor's degree in Computer Engineering from the Illinois Institute of Technology.

Contents

Foreword xxiii

Introduction xxv

Chapter 1 Introduction to Security and Firewalls 1

Introduction 2
The Importance of Security 2
What Is Information Security? 3
The Early Days of Information Security 5
Insecurity and the Internet 5
The Threats Grow 6
Attacks 7
Creating a Security Policy 8
Cisco's Security Wheel 11
Securing the Environment 12
Monitoring Activity 14
Testing Security 15
Improving Security 17
Firewall Concepts 17
What Is a Firewall? 17
Types of Firewalls 19
Packet Filters 20
Stateful Inspection Packet Filters 21
Application Proxies 22
Firewall Interfaces: Inside, Outside, and DMZ 23
Firewall Policies 26
Address Translation 26
Static Translation 27
Dynamic Translation 28
Port Address Translation 29

Virtual Private Networking 29
Cisco Security Certifications 31
Cisco Security Specialist 1 31
Requirements 32
Cisco Certified Internetwork Expert Security 32
The Written Test 33
The Lab Exam 33
CSPFA: The Exam 34
Exam Objectives 34
Summary 37
Solutions Fast Track 38
Frequently Asked Questions 40

Chapter 2 Introduction to PIX Firewalls 43
Introduction 44
PIX Firewall Features 44
Embedded Operating System 45
The Adaptive Security Algorithm 46
State 47
Security Levels 49
How ASA Works 49
Technical Details for ASA 50
User Datagram Protocol 54
Advanced Protocol Handling 55
VPN Support 56
URL Filtering 57
NAT and PAT 57
High Availability 59
PIX Hardware 59
Models 59
PIX 501 61
PIX 506 61
PIX 506E 61
PIX 515 61
PIX 515E 62
PIX 520 62
PIX 525 63
PIX 535 63

The Console Port 63
Software Licensing and Upgrades 65
Licensing 67
Upgrading Software 67
Password Recovery 69
The Command-Line Interface 71
Factory Default Configurations 71
PIX 501 and 506E 71
PIX 515E, 525, and 535 72
Administrative Access Modes 72
Basic Commands 75
Hostname and Domain Name 76
Configuring Interfaces 76
Static Routes 78
Password Configuration 78
Managing Configurations 79
The write Command 79
The copy Command 80
The configure Command 81
Resetting the System 82
The reload Command 82
Summary 83
Solutions Fast Track 85
Frequently Asked Questions 88

Chapter 3 Passing Traffic 91
Introduction 92
Allowing Outbound Traffic 92
Configuring Dynamic Address Translation 93
Identity NAT and NAT Bypass 97
Blocking Outbound Traffic 100
Access Lists 100
Outbound/Apply 109
Allowing Inbound Traffic 111
Static Address Translation 112
Access Lists 113
Conduits 113
ICMP 114

Port Redirection 115
TurboACLs 116
Object Grouping 117
Configuring and Using Object Groups 118
ICMP-Type Object Groups 118
Network Object Groups 119
Protocol Object Groups 119
Service Object Groups 120
Case Study 122
Access Lists 124
Conduits and Outbound/Apply 127
Summary 130
Solutions Fast Track 130
Frequently Asked Questions 132

Chapter 4 Advanced PIX Configurations 135

Introduction 136
Handling Advanced Protocols 136
File Transfer Protocol 141
Active vs. Passive Mode 141
Domain Name Service 146
Simple Mail Transfer Protocol 148
Hypertext Transfer Protocol 150
Remote Shell 150
Remote Procedure Call 152
Real-Time Streaming Protocol, NetShow, and VDO Live 153
SQL*Net 157
H.323 and Related Applications 159
Skinny Client Control Protocol 161
Session Initiation Protocol 162
Internet Locator Service and Lightweight Directory Access Protocol 164
Filtering Web Traffic 165
Filtering URLs 166
Websense and N2H2 167
Fine-Tuning and Monitoring the Filtering Process 169
Active Code Filtering 173
Filtering Java Applets 174
Filtering ActiveX Objects 174

Configuring Intrusion Detection 175
Supported Signatures 175
Configuring Auditing 179
Disabling Signatures 181
Configuring Shunning 182
DHCP Functionality 182
DHCP Clients 183
DHCP Servers 185
Cisco IP Phone-Related Options 189
Other Advanced Features 189
Fragmentation Guard 189
AAA Floodguard 191
SYN Floodguard 192
Reverse-Path Forwarding 194
Unicast Routing 197
Static and Connected Routes 197
Routing Information Protocol 199
Stub Multicast Routing 202
SMR Configuration with Clients on a More Secure Interface 204
SMR Configuration with Clients on a Less Secure Interface 206
Access Control and Other Options 207
PPPoE 209
Summary 212
Solutions Fast Track 213
Frequently Asked Questions 215

Chapter 5 Configuring Authentication, Authorization, and Accounting 217

Introduction 218
AAA Concepts 218
Authentication 221
Authorization 222
Accounting 223
AAA Protocols 223
RADIUS 223
TACACS+ 225

Cisco Secure ACS for Windows 228
Introduction and Features 229
Installing and Configuring Cisco Secure ACS 230
Adding an NAS to Cisco Secure ACS 237
Adding a User to Cisco Secure ACS 240
Configuring Console Authentication 242
Configuring Local Console Authentication 243
Configuring RADIUS and TACACS+ Console Authentication 244
Configuring TACACS+ Enable Console Authentication in Cisco Secure ACS 246
Configuring Command Authorization 250
Configuring Local Command Authorization 251
Configuring TACACS+ Command Authorization 252
Configuring Cisco Secure ACS to Support TACACS+ Command Authorization 253
Defining the Shell Command Authorization Set 255
Assigning the Command Authorization Set to Users or Groups 258
Enabling Command Authorization on the PIX Firewall 260
Configuring Authentication for Traffic Through the Firewall 260
Configuring Cut-Through Proxy 260
Virtual HTTP 266
Virtual Telnet 268
Configuring Authorization for Traffic Through the Firewall 270
Configuring Accounting for Traffic Through the Firewall 272
Configuring Downloadable Access Lists 275
Configuring Named Downloadable Access Lists 275
Configuring Downloadable Access Lists Without Names 280
Summary 282
Solutions Fast Track 283
Frequently Asked Questions 287

Chapter 6 Configuring System Management 289

Introduction 290
Configuring Logging 290
Local Logging 291
Buffered Logging 292

Console Logging 293
Terminal Logging 293
Syslog 293
Logging Levels 299
Logging Facility 302
Disabling Specific Syslog Messages 303
Configuring Remote Access 304
Secure Shell 305
Enabling SSH Access 306
Troubleshooting SSH 311
Telnet 314
Restrictions 315
HTTP Via the PIX Device Manager 316
Configuring Simple Network Management Protocol 316
Configuring System Identification 317
Configuring Polling 318
Configuring Traps 320
Configuring System Date and Time 321
Setting and Verifying the Clock and Time Zone 322
Configuring and Verifying the Network Time Protocol 324
NTP Authentication 325
Summary 327
Solutions Fast Track 328
Frequently Asked Questions 330

Chapter 7 Configuring Virtual Private Networking 333

Introduction 334
IPsec Concepts 334
IPsec 335
IPsec Core Layer 3 Protocols: ESP and AH 335
IPsec Communication Modes: Tunnel and Transport 338
Internet Key Exchange 340
Security Associations 343
Certificate Authority Support 348
Configuring Site-to-Site IPsec Using IKE 349
Planning 349
Allowing IPsec Traffic 350
Enabling IKE 352

Creating an ISAKMP Protection Suite 352
Defining an ISAKMP Pre-Shared Key 354
Configuring Certificate Authority Support 354
Configuring the Hostname and Domain Name 356
Generating an RSA Key Pair 356
Specifying a CA to Be Used 357
Configuring CA Parameters 358
Authenticating the CA 358
Enrolling with the CA 360
Configuring Crypto Access Lists 362
Defining a Transform Set 364
Bypassing Network Address Translation 365
Configuring a Crypto Map 366
Troubleshooting 369
Configuring Site-to-Site IPsec Without IKE (Manual IPsec) 369
Configuring Point-to-Point Tunneling Protocol 372
Overview 373
Configuration 375
Setting Up Windows 2000 Clients 380
Configuring Layer 2 Tunneling Protocol with IPsec 383
Overview 384
Dynamic Crypto Maps 384
Configuration 386
Setting Up the Windows 2000 Client 389
Configuring Support for the Cisco Software VPN Client 390
Mode Configuration 391
Extended Authentication 392
VPN Groups 394
Sample Configurations of PIX and VPN Clients 397
Summary 407
Solutions Fast Track 408
Frequently Asked Questions 410

Chapter 8 Configuring Failover 413

Introduction 414
Failover Concepts 414
Configuration Replication 417
IP and MAC Addresses Used for Failover 418

Failure Detection 419
Stateful Failover 420
Standard Failover Using a Failover Cable 422
Configuring and Enabling Failover 423
Monitoring Failover 430
Failing Back 432
Disabling Failover 433
LAN-Based Failover 434
Configuring and Enabling Failover 434
Monitoring Failover 440
Failing Back 443
Disabling Failover 443
Summary 444
Solutions Fast Track 444
Frequently Asked Questions 446

Chapter 9 PIX Device Manager 449

Introduction 450
Features, Limitations, and Requirements 450
Supported PIX Firewall Hardware and Software Versions 451
PIX Device Requirements 451
Requirements for a Host Running the PIX Device Management Client 452
PIX Device Manager Limitations 454
Installing, Configuring, and Launching PDM 455
Preparing for Installation 455
Installing or Upgrading PDM 455
Obtaining a DES Activation Key 456
Configuring the PIX Firewall For Network Connectivity 457
Installing a TFTP Server 457
Upgrading the PIX Firewall and Configuring the DES Activation Key 458
Installing or Upgrading PDM on the PIX device 458
Enabling and Disabling PDM 459
Launching PDM 460
Configuring the PIX Firewall Using PDM 466
Using the Startup Wizard 467

Configuring System Properties 474
The Interfaces Category 475
The Failover Category 476
The Routing Category 478
The DHCP Server Category 480
The PIX Administration Category 481
The Logging Category 490
The AAA Category 491
The URL Filtering Category 492
The Auto Update Category 494
The Intrusion Detection Category 495
The Advanced Category 497
The Multicast Category 498
The History Metrics Category 499
Maintaining Hosts and Networks 500
Configuring Translation Rules 505
Configuring Access Rules 512
Access Rules 513
AAA Rules 517
Filter Rules 518
Configuring VPN 519
Configuring a Site-to-Site VPN 521
Configuring for the Cisco Software VPN Client 525
Monitoring the PIX Firewall Using PDM 532
Sessions and Statistics 534
Graphs 537
VPN Connection Graphs 539
System Graphs 540
Connection Graphs 541
Miscellaneous Graphs 543
Interface Graphs 544
Monitoring and Disconnecting Sessions 547
Summary 548
Solutions Fast Track 549
Frequently Asked Questions 551

Chapter 10 Troubleshooting and Performance Monitoring 553
Introduction 554
Troubleshooting Hardware and Cabling 555
Troubleshooting PIX Hardware 556
Troubleshooting PIX Cabling 567
Troubleshooting Connectivity 570
Checking Addressing 571
Checking Routing 573
Checking Translation 580
Checking Access 583
Troubleshooting IPsec 588
IKE 591
IPsec 594
Capturing Traffic 597
Displaying Captured Traffic 599
Display on the Console 599
Display to a Web Browser 600
Downloading Captured Traffic 600
Monitoring and Troubleshooting Performance 602
CPU Performance Monitoring 604
The show cpu usage Command 605
The show processes Command 606
The show perfmon Command 608
Memory Performance Monitoring 609
The show memory Command 609
The show xlate Command 610
The show conn Command 610
The show block Command 610
Network Performance Monitoring 611
The show interface Command 611
The show traffic Command 612
Identification (IDENT) Protocol and PIX Performance 613
Summary 614
Solutions Fast Track 615
Frequently Asked Questions 617

Index 619

Foreword

As one of the first technologies employed to protect networks from unauthorized access, the firewall has come to exemplify network security. While an overall security strategy requires the harmonious integration of people, process, and technology to reduce risk, there is no doubt that firewalls can be a very valuable security tool when properly implemented. Today, the use of firewalls has become such an accepted practice that their deployment in one fashion or another is virtually a foregone conclusion when designing and building networks. Recognizing this need, Cisco Systems has developed and continues to improve upon its line of PIX firewalls. These systems have steadily gained market leadership by demonstrating an excellent mix of functionality, performance, and flexibility.

Firewalls have become increasingly sophisticated devices as the technology has matured. At its most basic level, a firewall is intended to enforce a security policy governing the network traffic that passes through it. To this basic functionality, Cisco has added many features such as network address translation (NAT), virtual private networks (VPN), and redundant architectures for high availability. Management systems are typically installed along with the firewall to assist with monitoring and administrating the device. A maxim of IT security is that technology is only as effective as the people responsible for its operation. Therefore, it is extremely important for the technical staff managing PIX firewalls to understand the technical functionality of these devices, as this will result in better security and more efficient operation of the equipment.

About This Book

The objective of this book is to provide you with a thorough understanding of the Cisco PIX firewalls. Whether you have administrative responsibilities or you are studying to pass an exam such as the Cisco Secure PIX Firewall Advanced (CPSFA), this comprehensive guide will be of value to you. The initial chapters cover the basics, and subsequent chapters delve into advanced topics. Callisma's contributing authors are industry experts with a wealth of real world implementation experience on the PIX and IOS firewalls, and this book includes many real-world examples of do's and don'ts. We hope you enjoy reading this book as much as we've enjoyed writing it!

—Ralph Troupe,
President and CEO, Callisma

About Callisma

Through Callisma's skilled team of technology, operations, and project management professionals, we enable today's major corporations to design and deploy networks that deliver business value. We help our clients compete effectively in the new e-business marketplace through strategic business planning, network design, and implementation services. By providing its clients with a broad base of technical services, a flexible, results-oriented engagement style, and the highest quality documentation and communication, Callisma delivers superior solutions—on time and on budget. Callisma's expertise includes IP Telephony, Internetworking, Storage, Optical Networking, Operations Management, Security, and Project Management. Callisma is headquartered in Silicon Valley, with offices located throughout the United States. For more information, visit the Callisma Web site at www.callisma.com or call 888.805.7075

Introduction

In an age when our society relies so heavily on electronic communication, the need for information security is imperative. Given the value and confidential nature of the information that exists on today's networks, CIOs are finding that an investment in security is not only extremely beneficial but also absolutely necessary. Corporations are realizing the need to create and enforce an information security policy. As a result, IT professionals are constantly being challenged to secure their networks by installing firewalls and creating Virtual Private Networks (VPNs) that provide secure, encrypted communications over the Internet's vulnerable public infrastructure.

Cisco's industry-leading PIX 500 Series firewall appliances (from the enterprise-class 535, to the plug-and-play SOHO model 501) deliver high levels of performance with unparalleled reliability, availability, and network security. With support for standards-based IPsec, VPNs, intrusion detection features, and a lot more, the PIX is one of the leading firewalls on the market.

Cisco Security Specialist's Guide to PIX Firewalls is a comprehensive guide for network and security engineers, covering the entire line of the PIX firewall product series. This book was written by highly experienced authors who provide high security solutions to their clients using Cisco PIX firewalls on a daily basis. This book covers all the latest and greatest features of PIX firewall software version 6.2, including TurboACLs, object grouping, NTP, HTTP failover replication, PIX Device Manager (PDM), and many others.

We have directed this book towards IT professionals who are preparing for the Cisco Secure PIX Firewall Advanced (CSPFA) written exam or the Cisco Certified Internet Expert (CCIE) Security written and lab exams. This book covers all the objectives of the CSPFA exam, and includes enough additional information to be useful to readers long after they have passed the exam. The content contained within these pages is useful to anyone who has a desire to fully comprehend Cisco PIX firewalls. This book serves as both a tool for learning and a reference guide. It is assumed

that the reader has a basic understanding of networking concepts and TCP/IP equivalent to that of a Cisco Certified Network Associated (CCNA). Here is a chapter-by-chapter breakdown of the book:

Chapter 1, "Introduction to Security and Firewalls," introduces general security and firewall concepts. For readers new to the area of information security, this chapter will guide them through fundamental security and firewall concepts that are necessary to understand the following chapters. The first and most important step towards starting to control network security is to establish a security policy for the company. The reader will learn how to create a security policy, and whom to involve when creating the policy. Information security is not a goal or a result; it is a process, and this is clearly demonstrated by the Cisco Security Wheel discussed in this chapter. Chapter 1 explains firewall concepts in detail, including the differences between the different types of firewalls, how firewalls work, and a look at firewall terminology. The chapter ends with a discussion of Cisco's security certifications and the objectives for the CSS-1 and CCIE Security written exams.

Chapter 2, "Introduction to PIX Firewalls," goes through the fundamentals of PIX firewalls. The main features of the PIX firewall are described, as well as the paradigm of PIX firewall configuration. The concepts of security levels and the Adaptive Security Algorithms (ASA), which are integral to PIX firewall operation, are also discussed in this chapter. The PIX firewall provides a scalable architecture with many different hardware offerings, designed to support SOHO in addition to enterprise and service provider environments. This chapter describes the various hardware models and introduces the PIX Command Line Interface (CLI). Basic commands that are needed to get the firewall up and running are included as well.

Chapter 3, "Passing Traffic," builds on the basic configuration information introduced in Chapter 2. Using a variety of examples and a complex case study, the reader will become familiar with the different methods of routing inbound and outbound traffic through the PIX firewall. The various forms of address translation methods are described in detail. This chapter also discusses both the legacy methods of passing traffic (conduit and outbound/apply commands), as well as the new and preferred method of using access lists.

Chapter 4, "Advanced PIX Configurations," explores various advanced PIX firewall topics, including the configuration of complex protocols that operate over multiple or dynamic ports. Another feature covered in this chapter is the ability of the PIX firewall to block specific Web traffic, Java, and ActiveX applications. This chapter also describes intrusion detection features of the firewall, DHCP client and server

functionality, and Reverse Path Forwarding (RPF), and finishes up with a discussion of advanced features by providing detailed information on PIX firewall multicast configuration.

Chapter 5, "Configuring Authentication, Authorization, and Accounting," takes the reader through the process of configuring user-level security. After introducing AAA concepts and protocols (RADIUS and TACACS+), this chapter describes in detail how the PIX firewall can be configured as an AAA client for controlling administrative access to the firewall itself and/or traffic that is passing through the firewall. The reader will also learn how to install and configure Cisco's AAA server, Cisco Secure Access Control Server for Windows.

Chapter 6, "Configuring System Management," discusses the various management and maintenance practices for the PIX firewall. Logging is integral to these practices not only for monitoring or troubleshooting; it is invaluable for measuring system performance, identifying potential network bottlenecks, and detecting potential security violations. Also covered in this chapter are lessons on how to enable and customize logging features, maximize the remote administration features of the PIX firewall (using both in-band management (SSH, Telnet, and HTTP), and out-of-band management (SNMP)), and provides details on how to set the system date and time and the Network Time Protocol (NTP).

Chapter 7, "Configuring Virtual Private Networking," explores site-to-site and remote access VPNs on the PIX firewall using IPsec, L2TP, and PPTP. This chapter dissects the complicated topic of VPNs into easy to understand pieces. Step-by-step examples are provided for configuration of site-to-site and remote access VPNs using manual IPsec, IPsec with IKE using pre-shared keys, and IPsec with IKE using digital certificates.

Chapter 8, "Configuring Failover," covers high availability configurations on the PIX firewall comprehensively. The PIX firewall provides a feature known as failover, which is used to set up a hot standby backup firewall in case the primary fails. In this chapter, the reader will learn not only how failover works, but also how to configure it. The various types of failover are discussed, including standard and LAN-based and stateless and stateful.

Chapter 9, "PIX Device Manager," looks at the Graphical User Interface (GUI) based administration features of the PIX firewall. While most of the book is focused around learning the Command Line Interface (CLI), the goal of this chapter is to show the reader how many of the functions explored throughout the book can also be performed through the PIX Device Manager (PDM) GUI. In this chapter, the

reader will learn how to use the PDM to install, configure, and maintain the PIX firewall.

Chapter 10, "Troubleshooting and Performing Monitoring," ties up a number of the concepts in the book by looking at both proactive maintenance and reactive troubleshooting for the PIX firewall. The OSI model is used as the basis for the organization of this chapter, and the range of topics includes hardware, Layer 2 connectivity, address translation, IPsec, and traffic captures. Firewall performance and health need to be monitored proactively, and this chapter discusses the practices that will ensure that the PIX firewall is operating as it should.

Our hope is that the readers of *Cisco Security Specialist's Guide to PIX Firewalls* will become masters of installing, configuring, maintaining, and troubleshooting PIX firewalls, in addition to being ready to take the CSPFA exam. After the exam, we hope this book will then serve as a comprehensive reference to PIX firewalls, and will become an important part of the collection of resources used to manage and maintain your security infrastructure. Whether using the book to obtain your CSS-1 or CCIE certification, or simply to enhance your knowledge and understanding of Cisco PIX firewalls, we are sure you will find the material contained in these pages very useful.

—Umer Khan, CCIE #7410, MCSE, SCSA, SCNA, CCA, SCE, CNX

Chapter 1

Introduction to Security and Firewalls

Solutions in this chapter:

- The Importance of Security
- Creating a Security Policy
- Cisco's Security Wheel
- Firewall Concepts
- Cisco Security Certifications

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

In an age where our society relies so heavily on electronic communication, the need for information security is constantly increasing. Given the value and confidential nature of the information that exists on today's networks, CIOs are finding that an investment in security is extremely beneficial. Without security, a company can suffer from theft or alteration of data, legal ramifications, and other issues that all result in monetary losses. Consequently, corporations are realizing the need to create and enforce an information security policy.

In this chapter, you will learn about why information security is necessary. We also look at how and why security policies are created and how security needs to be handled as a process. We look at firewalls in general, explore the different types of firewalls available in the market, and learn basic concepts about how firewalls work. Finally, we discuss the two main security certifications Cisco offers: the Cisco Security Specialist 1 (CSS-1) and the Cisco Certified Internet Expert (CCIE) Security.

The Importance of Security

Over the last couple of decades, many companies began to realize that their most valuable assets were not only their buildings or factories but also the intellectual property and other information that flowed internally as well as outwardly to suppliers and customers. Company managers, used to dealing with risk in their business activities, started to think about what might happen if their key business information fell into the wrong hands, perhaps a competitor's. For a while, this risk was not too large, due to how and where that information was stored. *Closed systems* was the operative phrase. Key business information, for the most part, was stored on servers accessed via terminals or terminal emulators and had few interconnections with other systems. Any interconnections tended to be over private leased lines to a select few locations, either internal to the company or to a trusted business partner.

However, over the last five to seven years, the Internet has changed how businesses operate, and there has been a huge acceleration in the interconnectedness of organizations, systems, and networks. Entire corporate networks have access to the Internet, often at multiple points. This proliferation has created risks to sensitive information and business-critical systems where they had barely existed before. The importance of information security in the business environment has now been underscored, as has the need for skilled, dedicated practitioners of this specialty.

What Is Information Security?

We have traditionally thought of security as consisting of people, sometimes with guns, watching over and guarding tangible assets such as a stack of money or a research lab. Maybe they sat at a desk and watched via closed-circuit cameras installed around the property. These people usually had minimal training and sometimes did not understand much about what they were guarding or why it was important. However, they did their jobs (and continue to do so) according to established processes, such as walking around the facility on a regular basis and looking for suspicious activity or people who do not appear to belong there.

Information security moves that model into the intangible realm. Fundamentally, information security involves making sure that only authorized people (and systems) have access to information. Information security professionals sometimes have different views on the role and definition of information security. One definition offered by Simson Garfinkel and Gene Spafford is, "A computer is secure if you can depend on it and its software to behave as you expect." This definition actually implies a lot. If information stored on your computer system is not there when you go to access it, or if you find that it has been tampered with, you can no longer depend on it as a basis for making business decisions. What about nonintrusive attacks, though—such as someone eavesdropping on a network segment and stealing information such as passwords? This definition does not cover that scenario, since nothing on the computer in question has changed. It is operating normally, and it functions as its users expect. Sun Microsystems' mantra of "The Network is the Computer" is true. Computing is no longer just what happens on a mainframe, a minicomputer, or a server; it also includes the networks that interconnect systems.

The three primary areas of concern in information security have traditionally been defined as follows:

- **Confidentiality** Ensuring that only authorized parties have access to information. Encryption is a commonly used tool to achieve confidentiality. Authentication and authorization, treated separately in the following discussion, also help with confidentiality.
- **Integrity** Ensuring that information is not modified by unauthorized parties (or even improperly modified by authorized ones!) and that it can be relied on. Checksums and hashes are used to validate data integrity, as are transaction-logging systems.

- **Availability** Ensuring that information is accessible when it is needed. In addition to simple backups of data, availability includes ensuring that systems remain accessible in the event of a denial of service (DoS) attack. Availability also means that critical data should be protected from erasure—for example, preventing the wipeout of data on your company's external Web site.

Often referred to simply by the acronym *CIA*, these three areas serve well as a security foundation. To fully scope the role of information security, however, we also need to add a few more areas of concern to the list. Some security practitioners include the following within the three areas described, but by getting more granular, we can get a better sense of the challenges that must be addressed:

- **Authentication** Ensuring that users are, in fact, who they say they are. Passwords, of course, are the longstanding way to authenticate users, but other methods such as cryptographic tokens and biometrics are also used.
- **Authorization/access control** Ensuring that a user, once authenticated, is only able to access information to which he or she has been granted permission by the owner of the information. This can be accomplished at the operating system level using file system access controls or at the network level using access controls on routers or firewalls.
- **Auditability** Ensuring that activity and transactions on a system or network can be monitored and logged in order to maintain system availability and detect unauthorized use. This process can take various forms: logging by the operating system, logging by a network device such as a router or firewall, or logging by an intrusion detection system (IDS) or packet-capture device.
- **Nonrepudiation** Ensuring that a person initiating a transaction is authenticated sufficiently such that he or she cannot reasonably deny that they were the initiating party. Public key cryptography is often used to support this effort.

You can say that your information is secure when all seven of these areas have been adequately addressed. The definition of *adequately* depends, however, on how much risk exists in each area. Some areas may present greater risk in a particular environment than in others.

The Early Days of Information Security

If we set the dial on our "way-back machine" to the 1980s, we would find that the world of information security was vastly different from today. Companies' "important" computing was performed on large, expensive systems that were tightly controlled and sat in very chilly rooms with limited human access. Users got their work done either via terminals connected to these large computers or large metal IBM PCs on their desks. These terminals pretty much allowed users to do only what the application and systems programmers enabled them to, via menus and perhaps a limited subset of commands to run jobs. Access control was straightforward and involved a small set of applications and their data, and frankly, not many users outside the glass room understood how to navigate around a system from a command prompt. As far as PCs were concerned, management's view was that nothing important was really happening with users' Lotus 1-2-3 spreadsheets, so they were not a security concern.

Networking was limited in extent. Corporate local area networks (LANs) were nearly nonexistent. Technologies such as X.25 and expensive leased lines at the then blazing speeds of 56kbps ruled the day. Wide area network (WAN) links were used to move data from office to office in larger companies, and sometimes to other related entities. Because networks consisted of a series of point-to-point private links, the risk of an intruder gaining access to inner systems was slim.

Insecurity and the Internet

The federation of networks that became the Internet consisted of a relatively small community of users by the 1980s, primarily in the research and academic communities. Because it was rather difficult to get access to these systems and the user communities were rather closely knit, security was not much of a concern in this environment, either. The main objective of connecting these various networks together was to share information, not keep it locked away. Technologies such as the UNIX operating system and the Transmission Control Protocol/Internet Protocol (TCP/IP) networking protocols that were designed for this environment reflected this lack of security concern. Security was simply viewed as unnecessary.

By the early 1990s, however, commercial interest in the Internet grew. These commercial interests had very different perspectives on security, ones often in opposition to those of academia. Commercial information had value, and access to it needed to be limited to specifically authorized people. UNIX, TCP/IP, and connections to the Internet became avenues of attack and did not have much

capability to implement and enforce confidentiality, integrity, and availability. As the Internet grew in commercial importance, with numerous companies connecting to it and even building entire business models around it, the need for increased security became quite acute. Connected organizations now faced threats that they had never had to consider before.

The Threats Grow

When the corporate computing environment was a closed and limited-access system, threats mostly came from inside the organizations. These *internal threats* came from disgruntled employees with privileged access who could cause a lot of damage. Attacks from the outside were not much of an issue since there were typically only a few, if any, private connections to trusted entities. Potential attackers were few in number, since the combination of necessary skills and malicious intent were not at all widespread.

With the growth of the Internet, *external threats* grew as well. There are now millions of hosts on the Internet as potential attack targets, which entice the now large numbers of attackers. This group has grown in size and skill over the years as its members share information on how to break into systems for both fun and profit. Geography no longer serves as an obstacle, either. You can be attacked from another continent thousands of miles away just as easily as from your own town.

Threats can be classified as structured or unstructured. *Unstructured threats* are from people with low skill and perseverance. These usually come from people called *script kiddies*—attackers who have little to no programming skill and very little system knowledge. Script kiddies tend to conduct attacks just for bragging rights among their groups, which are often linked only by an Internet Relay Chat (IRC) channel. They obtain attack tools that have been built by others with more skill and use them, often indiscriminately, to attempt to exploit a vulnerability on their target. If their attack fails, they will likely go elsewhere and keep trying. Additional risk comes from the fact that they often use these tools with little to no knowledge of the target environment, so attacks can wind up causing unintended results. Unstructured threats can cause significant damage or disruption, despite the attacker's lack of sophistication. These attacks are usually detectable with current security tools.

Structured attacks are a greater threat since they are conducted by skilled hackers who have a plan and a goal. If existing tools do not work for them, they simply modify them or write their own. They are able to discover new vulnerabilities in systems by executing complex actions that the system designers did not protect against. Structured attackers often use so-called *zero-day exploits*, which are

exploits that target vulnerabilities that the system vendor has not yet issued a patch for or does not even know about. Structured attacks often have stronger motivations behind them than simple mischief. These motivations or goals can include theft of source code, theft of credit card numbers for resale or fraud, retribution, or destruction or disruption of a competitor. A structured attack might not be blocked by traditional methods such as firewalls or detected by an IDS. It could even use non-computer methods such as social engineering.

NOTE

Social engineering, also known as *people hacking*, is a means for obtaining security information from people by tricking them. The classic example is calling up a user and pretending to be a system administrator. The hacker asks the user for his or her password to ostensibly perform some important maintenance task. To avoid being hacked via social engineering, educate your user community that they should always confirm the identity of any person calling them and that passwords should never be given to *anyone* over e-mail, instant messaging, or the phone.

Attacks

With the growth of the Internet, many organizations focused their security efforts on defending against outside attackers (that is, anyone originating from an external network) who are not authorized to access their systems. Firewalls were the primary focus of these efforts. Money was spent on building a strong perimeter defense, resulting in what Bill Cheswick from Bell Labs famously described years ago as "a crunchy shell around a soft, chewy center." Any attacker who succeeded in getting through (or around) the perimeter defenses would then have a relatively easy time compromising internal systems. This situation is analogous to the enemy parachuting into the castle keep instead of breaking through the walls (the technology is off by a few centuries, but you get the idea!). Perimeter defense is still vitally important, given the increased threat level from outside the network. However, it is simply no longer adequate by itself.

Various information security studies and surveys have found that the majority of attacks actually come from inside the organization. The internal threat can include authorized users attempting to exceed their permissions or unauthorized users trying to go where they should not be at all. The insider is potentially more dangerous than outsiders because he or she has a level of access that the outsider

does not—to both facilities and systems. Many organizations lack the internal preventive controls and other countermeasures to adequately defend against this threat. Networks are wide open, servers could be sitting in unsecured areas, system patches might be out of date, and system administrators might not review security logs.

The greatest threat, however, arises when an insider colludes with a structured outside attacker. The outsider's skills, combined with the insider's access, could result in substantial damage or loss to the organization.

Attacks can be defined in three main categories:

- **Reconnaissance attacks** Hackers attempt to discover systems and gather information. In most cases, these attacks are used to gather information to set up an access or a DoS attack. A typical reconnaissance attack might consist of a hacker pinging IP addresses to discover what is alive on a network. The hacker might then perform a port scan on the systems to see which applications are running as well as try to determine the operating system and version on a target machine.
- **Access attacks** An access attack is one in which an intruder attempts to gain unauthorized access to a system to retrieve information. Sometimes the attacker needs to gain access to a system by cracking passwords or using an exploit. At other times, the attacker already has access to the system but needs to escalate his or her privileges.
- **DoS attacks** Hackers use DoS attacks to disable or corrupt access to networks, systems, or services. The intent is to deny authorized or valid users access to these resources. DoS attacks typically involve running a script or a tool, and the attacker does not require access to the target system, only a means to reach it. In a distributed DoS (DDoS) attack, the source consists of many computers that are usually spread across a large geographic boundary.

Creating a Security Policy

A comprehensive security policy is fundamental to an effective information security program, providing a firm basis for all activities related to the protection of information assets. In creating their policies, organizations take one of two basic approaches: that which is not expressly prohibited is allowed, or that which is not explicitly allowed is prohibited. The chosen approach is usually reflective of the organization's overall culture.

Designing & Planning…

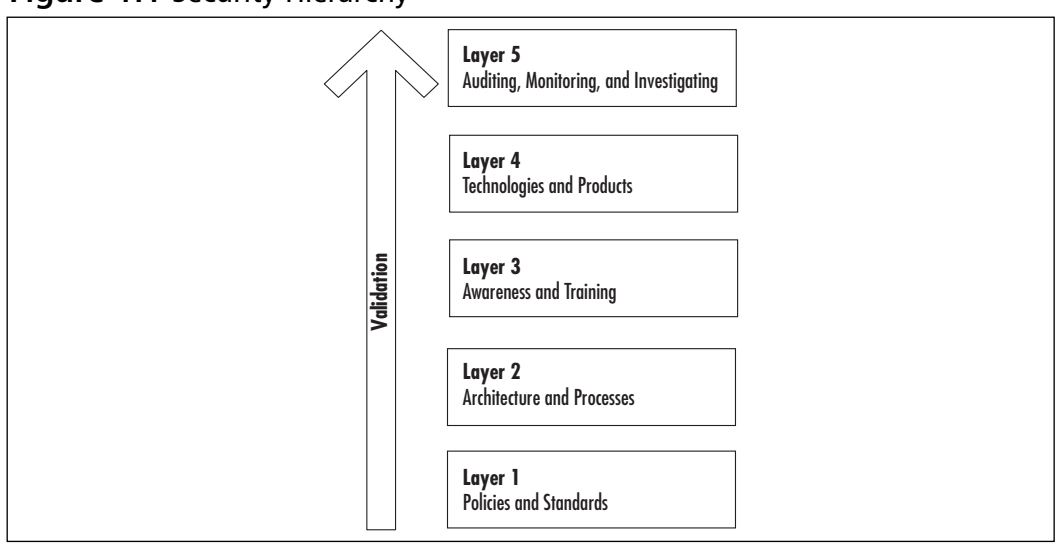
Developing a Comprehensive Security Policy

A good security policy addresses the following areas:

- Defines roles and responsibilities
- Defines acceptable use of the organization's computing resources
- Serves as a foundation for more specific procedures and standards
- Defines data sensitivity classifications
- Helps prevent security incidents by making clear management's expectations for protecting information
- Provides guidance in the event of a security incident
- Specifies results of noncompliance

Figure 1.1 shows a hierarchical security model. Each layer builds on the ones beneath it, with security policies serving as the foundation. An organization that implements security tools without defining good policies and architecture is likely to encounter difficulties.

Figure 1.1 Security Hierarchy



Creation of the security policy is guided by management's level of trust in the organization's people, de facto processes, and technology. Many organizations resist formalizing their policies and enforcing them, since they do not want to risk damaging their familial and trusting culture. When a security incident occurs, however, these organizations discover that they might have little or no guidance on how to handle it or that they do not have a legal foundation to prosecute or even terminate an employee who breaches security. Others follow a command-and-control model and find that defining policies fits right into their culture. These organizations, however, could wind up spending a great deal of money to enforce controls that provide little incremental reduction in risk and create an oppressive atmosphere that is not conducive to productivity. For most organizations, a middle approach is best, following the dictum "Trust, but verify."

The policy creation process might not be easy. People have very different ideas about what policies represent and why they are needed. The process should strive to achieve a compromise among the various stakeholders:

- Executive managers
- Internal auditors
- Human resources
- IT staff
- Security staff
- Legal staff
- Employee groups

As you can see, some level of buy-in from each of these stakeholder groups is necessary to create a successful policy. Particularly important is full support from executive management. Without it, a security policy will become just another manual gathering dust on the shelf. Employees need to see that management is behind the policy, leading by example.

Once a representative policy development team has been put together, its members should begin a risk-assessment process. The result of this effort is a document that defines how the organization approaches risk, how risk is mitigated, and the assets that are to be protected and their worth. The policy should also broadly define the potential threats that the organization faces. This information will be a guideline to the amount of effort and money that will be expended to address the threats and the level of risk that the organization will accept.

The next step is to perform a business needs analysis that defines information flows within the organization as well as information flowing into and out of it. These flows should each have a business need defined; this need is then matched with the level of risk to determine whether it will be allowed, allowed with additional controls, or restricted.

A good policy has these characteristics:

- States its purpose and what or who it covers
- Is realistic and easy to implement
- Has a long-term focus—in other words, does not contain specifics that will change often
- Is clear and concise
- Is up to date, with provisions for regular review
- Is communicated effectively to all affected parties, including regular awareness training
- Is balanced between security of assets and ease of use

Probably the most important component of a security policy is the definition of acceptable use. It covers how systems are to be used, user password practices, what users can and cannot do, user responsibility in maintaining security, and disciplinary action if users engage in improper activity. It is essential that all users sign this policy, acknowledging that they have read and understood it. Ideally, users should review the acceptable use policy on an annual basis. This practice helps reinforce the message that security is important.

Finally, an organization's security policy guides the creation of a perimeter security policy (including firewalls), which we cover in a later section.

NOTE

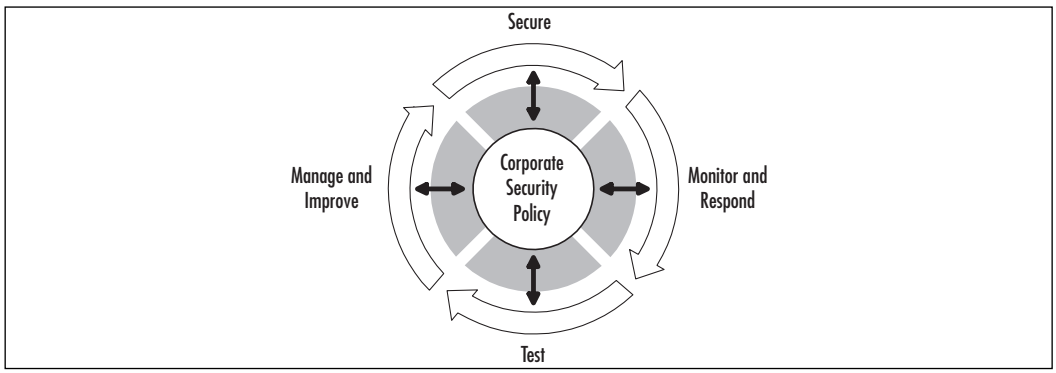
You'll find examples of security policies, including a sample acceptable use policy, on the *SANS Security Policy Resource* page located at www.sans.org/newlook/resources/policies.

Cisco's Security Wheel

Experienced security professionals often say that information security is not a goal or result, it is a process. This truism refers to the fact that you can never

secure your network and then be done with it. Information security is a dynamic field that is continually presenting challenges in the form of new technology, new threats, and new business processes. If you were to set a target secure state and then actually achieve it, you would find that the landscape had changed and further effort is required. One example of this sort of change is the ongoing discovery of vulnerabilities in existing software, for which patches must be applied. Although this process might seem daunting and often frustrating, it is what keeps many security practitioners interested in the field and excited about working in a mode of continuous improvement. Cisco has created a model, called the Cisco Security Wheel, that shows this process graphically (see Figure 1.2).

Figure 1.2 The Cisco Security Wheel



The Security Wheel really starts "rolling" when you have created your corporate security policy. The model defines four ongoing steps:

1. Secure the environment.
2. Monitor activity and respond to events and incidents.
3. Test the security of the environment.
4. Improve the security of the environment.

Each of these steps is discussed in detail in the following sections.

Securing the Environment

The task of securing an entire network can be overwhelming if viewed in the whole, especially if it covers multiple locations and thousands of systems. However, you can make the process much more manageable by breaking it down into smaller subtasks. Based on the risk analysis that was performed during the

policy development process, you can identify which of the following areas need attention first, second, and so on:

- **Confidentiality** For example, does your policy specify that sensitive information being communicated over public networks such as the Internet needs to be encrypted? If so, you might want to begin evaluating deployment of virtual private network (VPN) technology. A VPN creates an encrypted "tunnel" between two sites or between a remote user and the company network. Other efforts may include user education in handling of sensitive information.
- **Integrity** Does the risk assessment identify particular risks to company information? Does your company maintain a high-traffic Web site? Various tools and processes can be used to enhance the integrity of your information.
- **Availability** Various factors that have an impact on the availability of critical networks and systems might have been identified. This area of security, although important, will probably prove less critical than some of the others, unless you have been experiencing frequent system outages or have been the victim of frequent DoS attacks.
- **Authentication** Although it's one of the first lines of defense, authentication is a common area of weakness. Many organizations do not have adequate password policies and processes in place. For example, passwords are not changed on a regular basis, are not required to be of a certain level of complexity, or can be reused.
- **Access control** Another common area of weakness, access controls at both the network and system level, are often not as strong as they should be. Drives may be shared by all users with read/write access. The typical user has a greater level of access than he or she needs to do a job. Tightening up access controls can result in substantial improvements in a company's security posture. Some technological solutions include firewalls, router access lists, and policy enforcement tools that validate and perhaps control file system access.
- **Auditing** This is a primary activity in the next phase, monitoring.

Another key task in securing your systems is closing vulnerabilities by turning off unneeded services and bringing them up to date on patches. Services that have no defined business need present an additional possible avenue of attack and

are just another component that needs patch attention. Keeping patches current is actually one of the most important activities you can perform to protect yourself, yet it is one that many organizations neglect. The Code Red and Nimda worms of 2001 were successful primarily because so many systems had not been patched for the vulnerabilities they exploited, including multiple Microsoft Internet Information Server (IIS) and Microsoft Outlook vulnerabilities. Patching, especially when you have hundreds or even thousands of systems, can be a monumental task. However, by defining and documenting processes, using tools to assist in configuration management, subscribing to multiple vulnerability alert mailing lists, and prioritizing patches according to criticality, you can get a better handle on the job. One useful document to assist in this process has been published by the U.S. National Institute of Standards and Technology (NIST), which can be found at http://csrc.nist.gov/publications/nistpubs/800-40/sp800-40.pdf (800-40 is the document number). Patch sources for a few of the key operating systems are located at:

- Microsoft Windows: http://windowsupdate.microsoft.com
- Sun Solaris: http://sunsolve.sun.com
- Red Hat Linux: www.redhat.com/apps/support/resources
- Hewlett-Packard HP/UX: http://us-support.external.hp.com

Also important is having a complete understanding of your network topology and some of the key information flows within it as well as in and out of it. This understanding helps you define different zones of trust and highlights where rearchitecting the network in places might improve security—for example, by deploying additional firewalls internally or on your network perimeter.

Monitoring Activity

As you make efforts to secure your environment, you move into the next phase of information security: establishing better mechanisms for monitoring activity on your network and systems. Adequate monitoring is essential so that you can be alerted, for example, when a security breach has occurred, when internal users are trying to exceed their authority, or when hardware or software failures are having an impact on system availability. Effective monitoring has two components: turning on capabilities already present on your systems and implementing tools for additional visibility. The first component includes use of the auditing function built into:

- Operating systems such as administrator account access.
- Network devices, as in login failures and configuration changes.
- Applications, including auditing capability in the application as created by the vendor (for commercial software), as well as auditing added within a custom-developed application. Monitored events tend to be more transactional in nature, such as users trying to perform functions they are not authorized for.

Most systems have such auditing turned off by default, however, and require you to specifically enable it. Be careful not to turn on too much, since you will be overwhelmed with data and will wind up ignoring it. This "turn on and tune" methodology flows into the second component, which also includes deployment of tools such as IDS on networks and hosts.

In any environment that contains more than a few systems, performing manual reviews of system and audit logs, firewall logs, and IDS logs becomes an impossible and overwhelming task. Various tools (such as Swatch, at www.oit.ucsb.edu/~eta/swatch) can perform log reduction and alert only on important events.

Testing Security

It is far, far better to test your own security and find holes than for a hacker to find them for you. An effective security program includes regular vulnerability assessments and penetration testing as well as updates to your risk assessment when there are significant changes to the business or the technology. For example, initiating extranet links to business partners or starting to provide remote broadband access to employees should be accompanied by an updated risk profile that identifies the risks of the new activity and the component threats, prioritized by probability and severity. This testing identifies the components that need to be better secured and the level of effort required.

Things that need to be tested or checked for include:

- Security policy compliance, including things like password strength
- System patch levels
- Services running on systems
- Custom applications, particularly public-facing Web applications
- New servers added to the network
- Active modems that accept incoming calls

A multitude of tools, both freeware and commercial off-the-shelf tools, are available to perform security testing. Some freeware tools include:

- **Nmap (www.insecure.org/nmap)** Nmap is one of the most commonly used network and port scanning tools, used by hackers and security professionals alike. It also has the ability to "fingerprint" the operating system of the target host by analyzing the responses to different types of probes.
- **Nessus (www.nessus.org)** Nessus is a powerful, flexible vulnerability-scanning tool that can test different target platforms for known holes. It consists of a server process that is controlled by a separate graphical user interface (GUI). Each vulnerability is coded via a plug-in to the Nessus system, so new vulnerabilities can be added and tested for.
- **whisker (http://sourceforge.net/projects/whisker)** whisker is a collection of PERL scripts used to test Web server CGI scripts for vulnerabilities, a common point of attack in the Web environment.
- **Security Auditor's Research Assistant (www-arc.com/sara)** SARA is a third-generation UNIX-based security assessment tool based on the original SATAN. SARA interfaces with other tools such as nmap and Samba for enhanced functionality.
- **L0phtCrack (www.atstake.com/research/lc)** L0phtCrack is used to test (crack) Windows NT passwords. It is a good tool to look for weak passwords.

Commercial tools include:

- **ISS Internet Scanner (www.iss.net)** Internet Scanner is used to scan networks for vulnerabilities. ISS also makes scanners specifically for databases, host systems, and wireless networks.
- **Symantec Enterprise Security Manager (www.symantec.com)** ESM helps monitor for security policy compliance.
- **PentaSafe VigilEnt Security Manager (www.pentasafe.com)** VigilEnt assesses for vulnerabilities across an enterprise with easy-to-use reporting.

In addition to testing security yourself, it is good practice to bring in security experts that are skilled in vulnerability assessments and penetration testing. These experts (sometimes known as *ethical hackers*) conduct attacks in the same manner

as a hacker would, looking for any holes accessible from the outside. They are also able to conduct internal assessments to validate your security posture against industry best practices or standards such as the Common Criteria (http://csrc.nist.gov/cc/) or ISO17799. Internal assessments include interviews with key staff and management, reviews of documentation, and testing of technical controls. A third-party review potentially provides a much more objective view of the state of your security environment and can even be useful in convincing upper management to increase IT security funding.

Improving Security

The fourth phase in the Security Wheel is that of improving security. In addition to securing your network, setting up monitoring, and performing vulnerability testing, you need to stay abreast, on a weekly or even daily basis, of current security news, primarily consisting of new vulnerability reports. Waiting for a particular vendor to alert you to new vulnerabilities is not enough; you also need to subscribe to third-party mailing lists such as Bugtraq (www.securityfocus.com) or Security Wire Digest (www.infosecuritymag.com). Also important is verifying configurations on key security systems on a regular basis to ensure that they continue to represent your current policy. Most important of all, the four steps of the Security Wheel must be repeated continuously.

Firewall Concepts

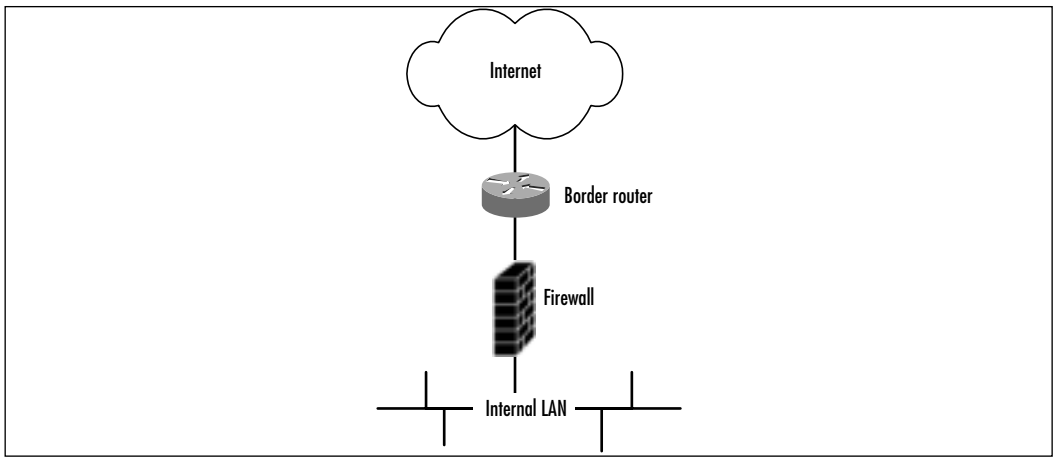
In this section, we discuss the concept and definition of firewalls and look at the different types of firewalls and some other architectural aspects such as network interfaces, address translation, and VPNs.

What Is a Firewall?

The term *firewall* comes from the bricks-and-mortar architectural world. In buildings, a firewall is a wall built from heat- or fire-resistant material such as concrete that is intended to slow the spread of fire through a structure. In the same way, on a network a firewall is intended to stop unauthorized traffic from traveling from one network to another. The most common deployment of firewalls occurs between a trusted network and an untrusted one, typically the Internet. Figure 1.3 depicts this configuration and shows the border router that terminates a serial connection from the Internet service provider (ISP). In the past, it was actually rather common for Internet-connected organizations to have no firewalls, instead simply relying on the security of their host systems to protect

their data. As networks got larger, it became unwieldy and risky to try to adequately secure each and every host, especially given the ever-increasing hacker threat.

Figure 1.3 Typical Firewall Placement



More and more sites, however, are also deploying firewalls into their internal networks, to separate zones of criticality. One example is putting a firewall between the payroll department subnet and the rest of the organization's network. In this case, the company security policy could have specified that the payroll data and systems are sensitive, that few (if any) employees outside the department need to initiate connections into it, and that payroll employees need outbound access to other local network resources as well as the Internet.

Firewall systems have certainly evolved over the years. Originally, firewalls were hand-built systems with two network interfaces that forwarded traffic between them. However, this was an area for experts only, requiring significant programming skills and system administration talent. Recognizing a need in this area, the first somewhat commercial firewall was written by Marcus Ranum (working for TIS at the time) in the early 1990s. It was called the Firewall Toolkit, or fwtk for short. It was an application proxy design (definitions of firewall types are in the following section) that intermediated network connections from users to servers. The goal was to simplify development and deployment of firewalls and minimize the amount of custom firewall building that would otherwise be necessary. The now familiar Gauntlet firewall product evolved from the original fwtk, and TIS was acquired by Network Associates, Inc. Other vendors got into the firewall market, including Check Point, Secure Computing, Symantec, and of course, Cisco.

Configuring & Implementing…

Deploying a Firewall

For quite some time, it was common for companies to think that once they deployed a firewall, they were secure. However, firewalls are just one component in an enterprise security strategy. They are generally good at what they do (filtering traffic), but they cannot do everything. The nature of perimeter security has also changed; many companies no longer need outbound-only traffic. Many enterprises now deal with much more complex environments that include business partner connections, VPNs, and complicated e-commerce infrastructures. This complexity has driven huge increases in firewall functionality. Most firewalls now support multiple network interfaces and can control traffic between them, support VPNs, and enable secure use of complicated application protocols such as H.323 for videoconferencing. The risk, however, is that as more and more functionality is added to the firewall, holes might arise in these features, compromising integrity and security. Another risk is that these features will exact a performance penalty, reducing the firewall's ability to focus on traffic filtering.

So the message is this: Try to use your firewall to the minimum extent possible so it can focus on its core function, and you can better manage the security risk of the other functions by shifting them to other systems to handle the load.

RBC Capital Markets estimated in a 2002 study that in 2000 the firewall market globally represented US$736 million, with an annual growth rate of 16 percent over the following five years. This shows that not everyone has deployed a firewall yet, that more companies are deploying them internally, and that there is ongoing replacement activity.

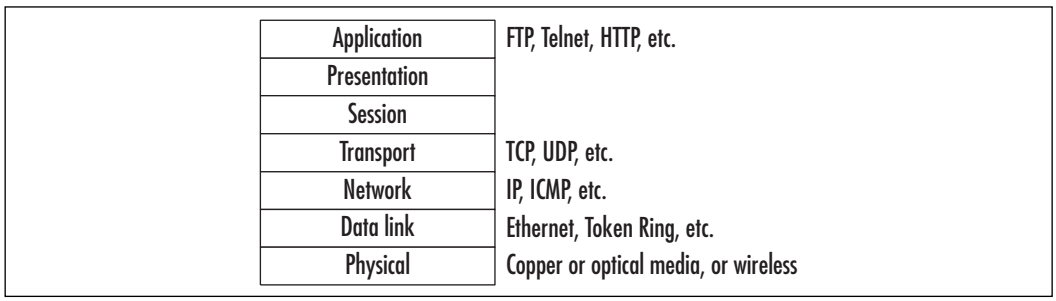
Next, let's look at the types of firewalls and compare their functionalities.

Types of Firewalls

Although the original fwtk used a proxy-type design, other types of firewalls use a much different approach. Before we look at these, recall the Open Systems Interconnect (OSI) model (see Figure 1.4).

Using this model as a reference, we can compare how the types of firewalls operate and make informed decisions about which type of firewall is appropriate for a particular need.

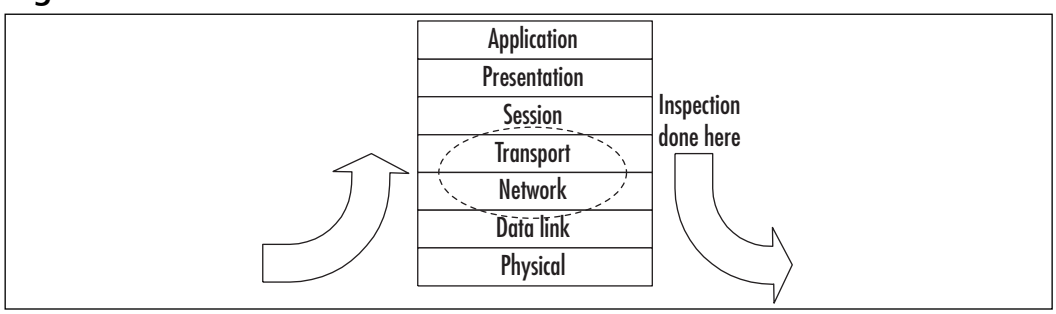
Figure 1.4 The OSI Model



Packet Filters

In its most basic form, a *packet filter* makes decisions about whether to forward a packet based only on information found at the IP or TCP/UDP layers; in effect, a packet filter is a router with some intelligence. However, a packet filter only handles each packet individually; it does not keep track of TCP sessions. Thus, it is poorly equipped to detect spoofed packets that come in through the outside interface, pretending to be part of an existing session by setting the ACK flag in the TCP header. Packet filters are configured to allow or block traffic according to source and destination IP addresses, source and destination ports, and type of protocol (TCP, UDP, ICMP, and so on). Figure 1.5 shows how inspection only goes as far as the transport layer—for example, TCP.

Figure 1.5 Packet Filter Data Flow



So why would you use a packet filter? The primary benefit is speed. Since it does not have to do any inspection of application data, a packet filter can operate nearly as fast as a router that is performing only packet routing and forwarding. As we will see, however, the packet filter concept has been improved.

Developing & Deploying…

Spoofing

The term *source address spoofing* refers to an attacker deliberately modifying the source IP address of a packet in an effort to trick packet filters or firewalls into thinking that the packet came from a trusted network so that it will pass the packet through. It also serves the obvious benefit of hiding the source of the attack packets. The attacker can also undermine any access controls that are based solely on the source IP address. If the source IP used is that of an existing host, however, the real owner of that address will receive any replies to the attacker's packets and will reject them with a TCP reset, since they do not match an existing session in its tables. An attacker will typically use spoofing when he or she just wants to initiate some action without needing to see a reply, as in a reflection DoS attack such as smurf, where a ping is sent to a broadcast address using the source IP of the intended DoS target.

More complicated attacks using IP spoofing are possible, particularly where the attacker is trying to exploit UNIX trust relationships. This is how Kevin Mitnick attacked Tsutomu Shimomura's systems on Christmas Day, 1994. Although Mitnick succeeded in his attack while coming over the Internet, this type of spoofing attack only works on an internal network these days (unless the victim has no firewall and is running old software).

Stateful Inspection Packet Filters

The concept of *stateful inspection* came about in an effort to improve on the capability and security of regular packet filters while still capitalizing on their inherent speed. A packet filter with stateful inspection is able to keep track of network sessions, so when it receives an ACK packet, it can determine its legitimacy by matching the packet to the corresponding entry in the connection table. An entry is created in the connection table when the firewall sees the first SYN packet that begins the TCP session. This entry is then looked up for succeeding packets in the session. Entries are automatically timed out after some configurable timeout period.

Statefulness can also be applied to UDP communication in a pseudo fashion, which normally has no concept of state. In this case, the firewall creates an entry in the connection table when the first UDP packet is transmitted. A UDP packet

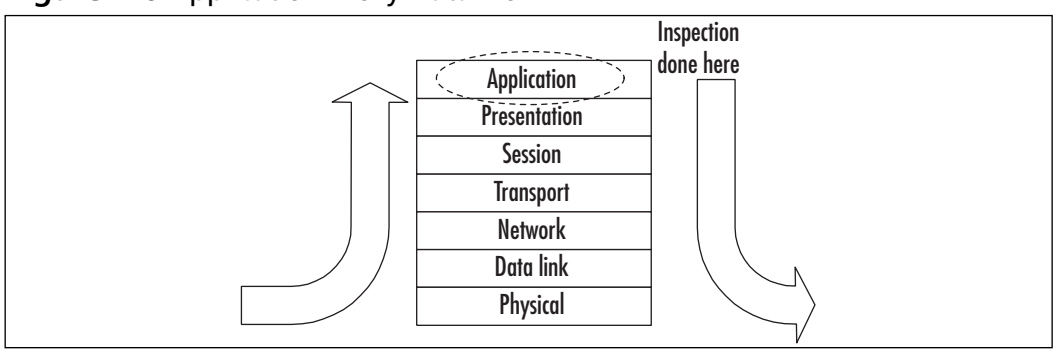
from a less secure network (a response) will only be accepted if a corresponding entry is found in the connection table. If we move up to the application layer, we can see further use for statefulness for protocols such as FTP. FTP is a bit different in that the server that the user connects to on port 21 will initiate a data connection back on port 20 when a file download is requested. If the firewall has not kept track of the FTP control connection that was initially established, it will not allow the data connection back in. This concept also applies to many of the newer multimedia protocols such as RealAudio and NetMeeting.

Stateful inspection packet filters remain the speed kings of firewalls and are the most flexible where new protocols are concerned, but they are sometimes less secure than application proxies. Check Point FireWall-1 and the Cisco PIX are the leading examples of this type of firewall.

Application Proxies

As their name implies, application proxy firewalls act as intermediaries in network sessions. The user's connection terminates at the proxy, and a corresponding separate connection is initiated from the proxy to the destination host. Connections are analyzed all the way up to the application layer to determine if they are allowed. It is this characteristic that gives proxies a higher level of security than packet filters, stateful or otherwise. However, as you might imagine, this additional processing extracts a toll on performance. Figure 1.6 shows how packet processing is handled at the application layer before it is passed on or blocked.

Figure 1.6 Application Proxy Data Flow



One potentially significant limitation of application proxies is that as new application protocols are implemented, corresponding proxies must be developed to handle them. This means that you could be at the mercy of your vendor if there is a hot new video multicasting technology, for example, but there is no proxy for it.

NOTE

Modern proxy-based firewalls often provide the ability to configure generic proxies for IP, TCP, and UDP. Although not as secure as proxies that work at the application layer, these configurable proxies often allow for passing of newer protocols.

Examples of proxy-based firewalls include Gauntlet from Secure Computing (acquired from Network Associates) and Symantec Raptor (also known as Enterprise Firewall).

Firewall Interfaces: Inside, Outside, and DMZ

In its most basic form, a firewall has just two network interfaces: inside and outside. These labels refer to the level of trust in the attached network, where the outside interface is connected to the untrusted network (often the Internet) and the inside interface is connected to the trusted network. In an internal deployment, the interface referred to as outside may be connected to the company backbone, which is probably not as untrusted as the Internet but just the same is trusted somewhat less than the inside. Recall the previous example of a firewall deployed to protect a payroll department.

As a company's Internet business needs become more complex, the limitations of having only two interfaces becomes apparent. For example, where would you put a Web server for your customers? If you place it on the outside of the firewall, as in Figure 1.7, the Web server is fully exposed to attacks, with only a screening router for minimal protection. You must rely on the security of the host system in this instance.

The other possibility in the two-interface firewall scenario is to put the Web server inside the firewall, on an internal segment (see Figure 1.8). The firewall would be configured to allow Web traffic on port 80, and maybe 443 for Secure Sockets Layer (SSL), through to the IP address of the Web server. This prevents any direct probing of your internal network by an attacker, but what if he or she is able to compromise your Web server through port 80 and gain remote superuser access? Then he or she is free to launch attacks from the Web server to anywhere else in your internal network, with no restrictions.

Figure 1.7 A Web Server Located Outside the Firewall

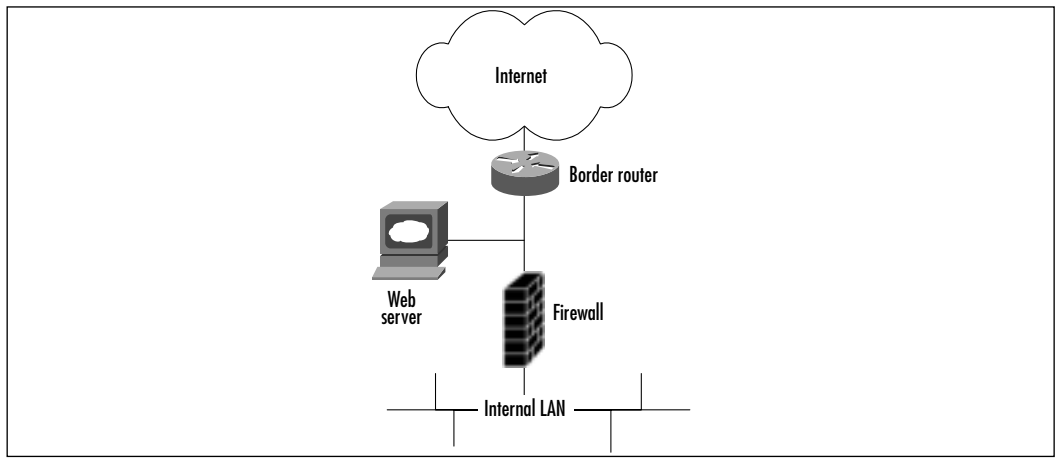
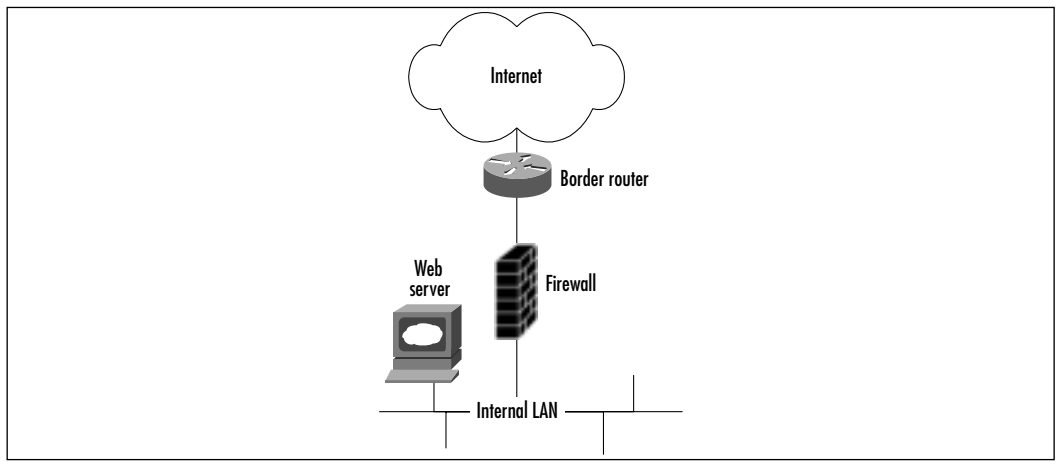
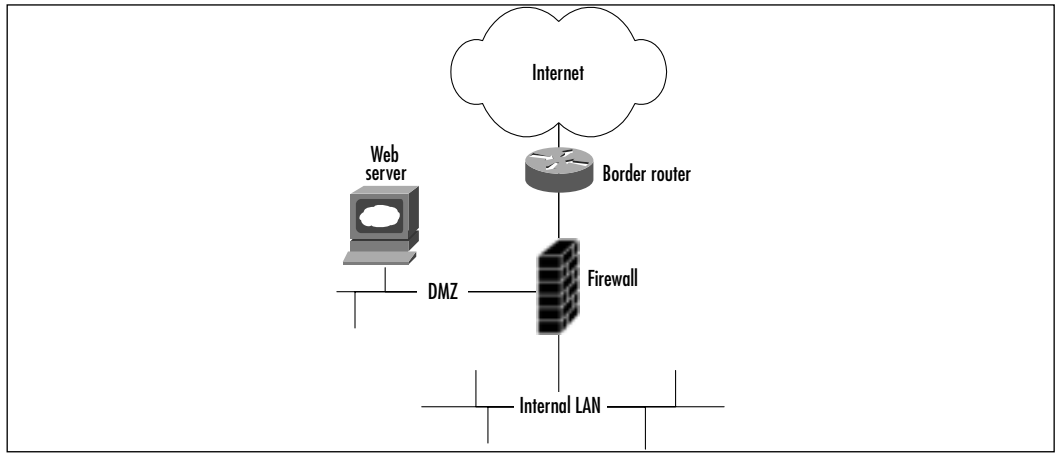


Figure 1.8 A Web Server Located Inside the Firewall



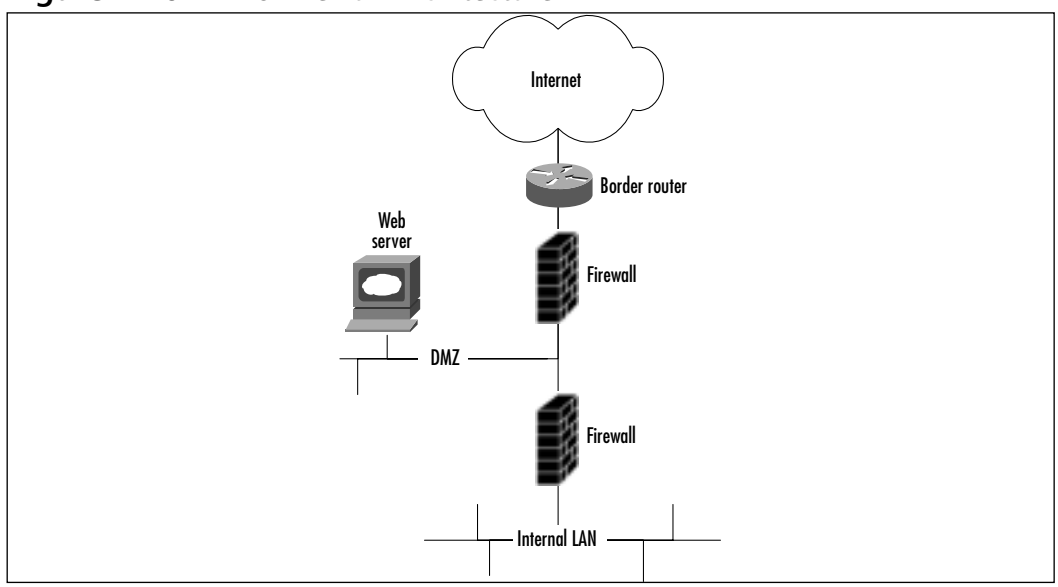
The answer to these problems is to have support for multiple interfaces on your firewall, as most commercial systems now do. This solution allows for establishment of intermediate zones of trust that are neither inside nor outside. These are referred to as DMZs (for the military term *demilitarized zone*). A DMZ network is protected by the firewall to the same extent as the internal network but is separated so that access from the DMZ to the internal network is filtered as well. Figure 1.9 shows this layout.

Figure 1.9 A DMZ Network



Another design sometimes deployed uses two firewalls: an outer one and an inner one, with the DMZ lying between them (see Figure 1.10). Sometimes firewalls from two different vendors are used in this design, with the belief that a security hole in one would be blocked by the other. However, evidence shows that nearly all firewall breaches come from misconfiguration, not from errors in the firewall code itself. Thus, such a design only increases expense and management overhead, without providing much additional security, if any.

Figure 1.10 A Two-Firewall Architecture



Some sites have even implemented multiple DMZs, each with a different business purpose and corresponding level of trust. For example, one DMZ segment could contain only servers for public access, whereas another could host servers just for business partners or customers. This approach enables a more granular level of control and simplifies administration.

In a more complex e-commerce environment, the Web server might need to access customer data from a backend database server on the internal LAN. In this case, the firewall would be configured to allow Hypertext Transfer Protocol (HTTP) connections from the outside to the Web server and then specific connections to the appropriate IP addresses and ports as needed from the Web server to the inside database server.

Firewall Policies

As part of your security assessment process, you should have a clear idea of the various business reasons for the different communications allowed through your firewall. Each protocol carries with it certain risks, some far more than others. These risks must be balanced with their business benefits. For example, one person needing X Windows (a notoriously difficult protocol to secure properly) access through the firewall for a university class she is taking is unlikely to satisfy this requirement. On the other hand, a drop-box File Transfer Protocol (FTP) server for sharing of files with customers might satisfy it. It often happens that the firewall rule base grows organically over time and reaches a point where the administrator no longer fully understands the reasons for everything in there. For that reason, it is essential that the firewall policy be well documented, with the business justification for each rule clearly articulated in this documentation. Changes to the firewall policy should be made sparingly and cautiously, only with management approval, and through standard system maintenance and change control processes.

Address Translation

RFC 1918, "Address Allocation for Private Internets," specifies certain nonregistered IP address ranges that are to be used only on private networks and are not to be routed across the Internet. The RFC uses the term *ambiguous* to refer to these private addresses, meaning that they are not globally unique. The reserved ranges are:

```
10.0.0.0        -   10.255.255.255  (10/8 prefix)
172.16.0.0      -   172.31.255.255  (172.16/12 prefix)
192.168.0.0     -   192.168.255.255 (192.168/16 prefix)
```

The primary motivation for setting aside these private address ranges was the fear in 1996 that the 32-bit address space of IP version 4 was becoming rapidly depleted due to inefficient allocation. Organizations that had at most a few thousand hosts, most of which did not need to be accessible from the Internet, over the years had been allocated huge blocks of IP addresses that had gone mostly unused. By renumbering their private networks with these reserved address ranges, companies could potentially return their allocated public blocks for use elsewhere, thus extending the useful life of IP v4.

The sharp reader, however, will point out that if these addresses are not routable on the Internet, how does one on a private network access the Web? The source IP of such a connection would be a private address, and the user's connection attempt would just be dropped before it got very far. This is where Network Address Translation (NAT), defined in RFC 1631, comes into play. Most organizations connected to the Internet use NAT to hide their internal addresses from the global Internet. This serves as a basic security measure that can make it a bit more difficult for an external attacker to map out the internal network. NAT is typically performed on the Internet firewall and takes two forms, static or dynamic. When NAT is performed, the firewall rewrites the source and/or the destination addresses in the IP header, replacing them with translated addresses. This process is configurable. First, some terms need to be defined. In the context of address translation, *inside* refers to the internal, private network. *Outside* is the greater network to which the private network connects (typically the Internet). Within the inside address space, addresses are referred to as *inside local* (typically RFC 1918 ranges) and are translated to *inside global* addresses that are visible on the outside. *Global* addresses are registered and assigned in blocks by an ISP. For translations of *outside* addresses coming to the *inside*, distinction is made also between *local*, part of the private address pool, and *global* registered addresses. *Outside local*, as the name might imply, is the reverse of inside global. These are addresses of outside hosts that are translated for access internally. *Outside global* addresses are owned by and assigned to hosts on the external network.

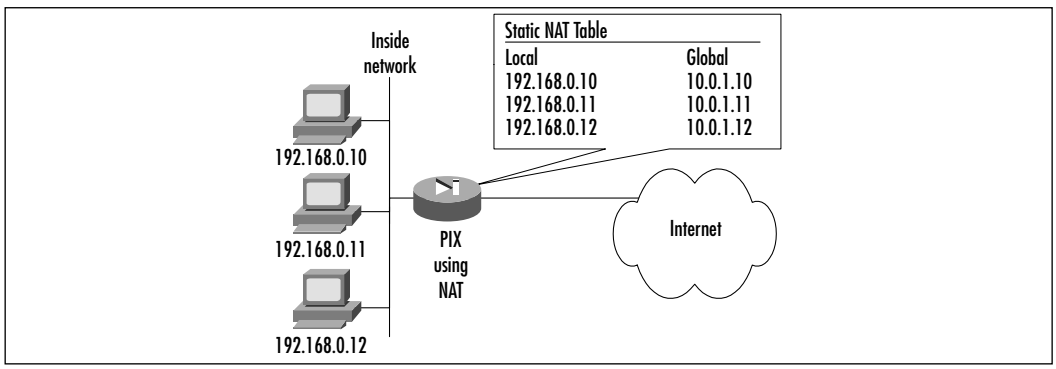
To keep these terms straight, just keep in mind the direction in which the traffic is going—in other words, from where it is initiated. This direction determines which translation will be applied.

Static Translation

In static NAT, a permanent one-to-one mapping is established between inside local and inside global addresses. This method is useful when you have a small number of inside hosts that need access to the Internet and have adequate

globally unique addresses to translate to. When a NAT router or firewall receives a packet from an inside host, it looks to see if there is a matching source address entry in its static NAT table. If there is, it replaces the local source address with a global source address and forwards the packet. Replies from the outside destination host are simply translated in reverse and routed onto the inside network. Static translation is also useful for outside communication initiated to an inside host. In this situation, the destination (not the source) address is translated. Figure 1.11 shows an example of static NAT. Each local inside address (192.168.0.10, 192.168.0.11, and 192.168.0.12) has a matching global inside address (10.0.1.10, 10.0.1.11, and 10.0.1.12, respectively).

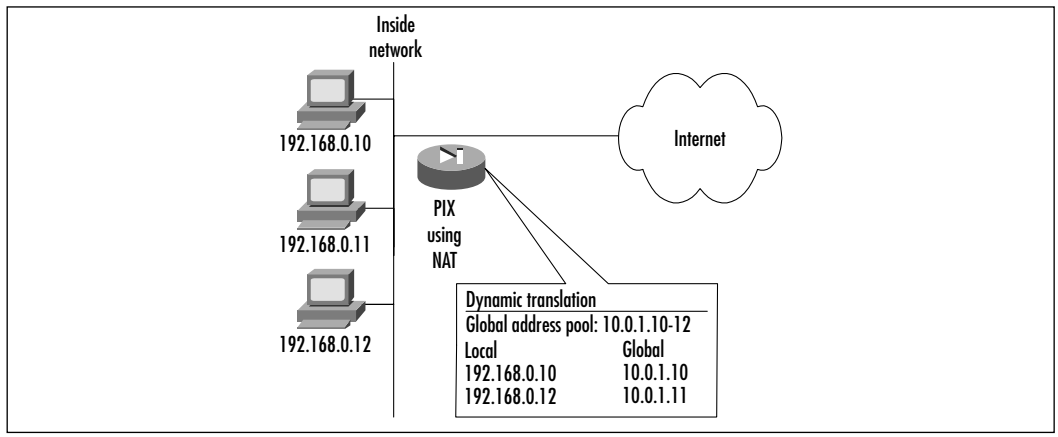
Figure 1.11 Static Address Translation



Dynamic Translation

When dynamic NAT is set up, a pool of inside global addresses is defined for use in outbound translation. When the NAT router or firewall receives a packet from an inside host and dynamic NAT is configured, it selects the next available address from the global address pool that was set up and replaces the source address in the IP header. Dynamic NAT differs from static NAT because address mappings can change for each new conversation that is set up between two given endpoints. Figure 1.12 shows how dynamic translation might work. The global address pool (for example purposes only) is 10.0.1.10 through 10.0.1.12, using a 24-bit subnet mask (255.255.255.0). The local address 192.168.0.10 is mapped directly to the first address in the global pool (10.0.1.10). The next system needing access (local address 192.168.0.12 in this example) is mapped to the next available global address of 10.0.1.11. The local host 192.168.0.11 never initiated a connection to the Internet, and therefore a dynamic translation entry was never created for it.

Figure 1.12 Dynamic Address Translation



Port Address Translation

What happens when there are more internal hosts initiating sessions than there are global addresses in the pool? This is called *overloading*, a configurable parameter in NAT, also referred to as *Port Address Translation,* or *PAT*. In this situation, you have the possibility of multiple inside hosts being assigned to the same global source address. The NAT/PAT box needs a way to keep track of which local address to send replies back to. This is done by using unique source port numbers as the tracking mechanism and involves possible rewriting of the source port in the packet header. You should recall that TCP/UDP uses 16 bits to encode port numbers, which allows for 65,536 different services or sources to be identified. When performing translation, PAT tries to use the original source port number if it is not already used. If it is, the next available port number from the appropriate group is used. Once the available port numbers are exhausted, the process starts again using the next available IP address from the pool.

Virtual Private Networking

The concept of VPN developed as a solution to the high cost of dedicated lines between sites that needed to exchange sensitive information. As the name indicates, it is not quite private networking, but "virtually private." This privacy of communication over a public network such as the Internet is typically achieved using encryption technology and usually addresses the issues of confidentiality, integrity, and authentication.

In the past, organizations that had to enable data communication between multiple sites used a variety of pricey WAN technologies such as point-to-point leased lines, Frame Relay, X.25, and Integrated Services Digital Network (ISDN). These were especially expensive for companies that had international locations. However, whether circuit-switched or packet-switched, these technologies carried an inherent decent measure of security. A hacker would typically need to get access to the underlying telecom infrastructure to be able to snoop on communications. This was, and still is, a nontrivial task, since carriers have typically done a good job on physical security. Even so, organizations such as banks that had extreme requirements for WAN security would deploy link encryption devices to scramble all data traveling across these connections. Another benefit to having dedicated links has been that you had a solid baseline of bandwidth that you could count on. Applications that had critical network throughput requirements would drive the specification of the size of WAN pipe that was needed to support them. VPNs experienced slow initial adoption due to the lack of throughput and reliability guarantees on the Internet as well as the complexity of configuration and management.

Now that the Internet has proven its reliability for critical tasks and many of the management hurdles have been overcome, VPN adopters are now focusing their attention on issues of interoperability and security. The interoperability question has mostly been answered as VPN vendors are implementing industry-standard protocols such as IPsec for their products. The IPsec standards provide for confidentiality, integrity, and optionally, authentication.

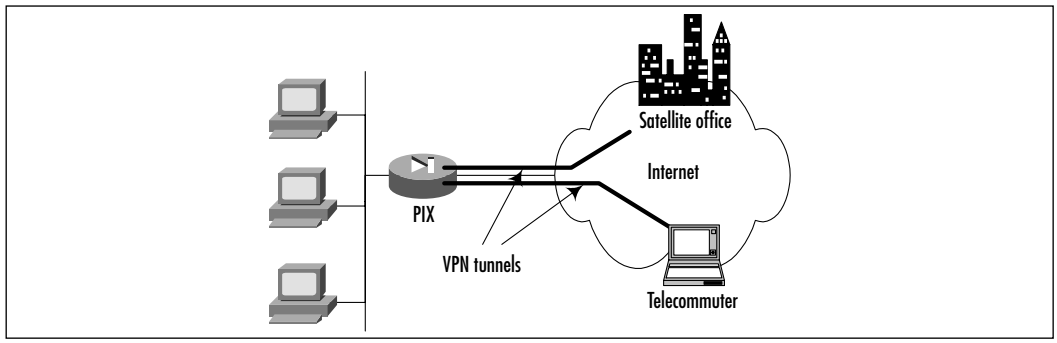
Security Alert

Many organizations have gone through the trouble of setting up VPN links for their remote users but have not taken the extra step of validating or improving the security of the computers that these workers are using to access the VPN. The most secure VPN tunnel offers no protection if the user's PC has been compromised by a Trojan horse program that allows a hacker to ride through the VPN tunnel right alongside legitimate, authorized traffic.

The solution is to deploy cost-effective firewall and intrusion detection software or hardware for each client that will be accessing the VPN, as well as continuous monitoring of the datastream coming out of the tunnel. Combined with real-time antivirus scanning and regular security scans, this solution helps ensure that the VPN does not become an avenue for attack into the enterprise.

Because of these improvements, organizations are now able to deploy VPNs in a rather straightforward manner, enabling secure access to the enterprise network for remote offices and/or telecommuters. Figure 1.13 shows the two main reasons for setting up VPNs. The first is to provide site-to-site connectivity to remote offices. The second is for telecommuters, adding flexibility by enabling enterprise access not only via dial-up to any ISP but also through a broadband connection via a home or hotel, for example. VPNs are used for many other reasons nowadays, including setting up connectivity to customers, vendors, and partners.

Figure 1.13 VPN Deployment



Cisco Security Certifications

Cisco has two certification paths for the practitioner to demonstrate competence in Cisco security technologies: Cisco Security Specialist 1 (CSS-1) and Cisco Certified Internetwork Expert (CCIE) Security. These two certifications show that the holder has significant experience and skills using and integrating Cisco security products, including VPN devices, IDS, and, of course, PIX firewalls.

Cisco Security Specialist 1

The CSS-1 certification is one of Cisco's Qualified Specialist designations. A person who has achieved the CSS-1 certification has proven through examination that he or she possesses a keen understanding of network security processes, technologies, and risks. He or she also understands how to deploy, configure, and manage Cisco security tools to support efforts in perimeter defense, network and host intrusion monitoring, and network-level encryption.

Requirements

The initial requirement to obtain the CSS-1 certification is a current Cisco Certified Network Associate (CCNA) certification. With that, the candidate can choose to get specific training through a Cisco Training Partner or Cisco e-learning to augment and reinforce their skills or simply sit in for the necessary written exams. There is no requirement that the candidate go through training in order to take the exams. However, because the exams are quite rigorous, the candidate should ensure that they meet all the knowledge objectives as described for each course and corresponding exam.

The current four exams that must be passed to obtain CSS-1 certification are shown in Table 1.1.

Table 1.1 CSS-1 Certification Exam Requirements

Exam Number	Training Course
640-100	Managing Cisco Network Security (MCNS)
9E0-111	Cisco Secure PIX Firewall Advanced (CSPFA)
9E0-572	Cisco Secure IDS with Policy Manager (IDSPM)
9E0-121	Cisco Secure VPN (CSVPN)

NOTE

Cisco keeps its certifications up to date; therefore, the certification requirements are constantly changing. Visit Cisco's Web site for the latest information on active exams.

A person with CSS-1 certification needs to recertify every two years by taking a written exam. Note that CSS-1 may be a requirement for certain Cisco partners to get and maintain their VPN/Security specialization.

Cisco Certified Internetwork Expert Security

The CCIE certification demonstrates that the holder belongs to the top tier of internetworking talent. The extremely challenging path to CCIE certification requires passing both a written test and a comprehensive hands-on lab exam. As an adjunct to the CCIE program, Cisco has created a security designation for

those who want to demonstrate additional top-level competence in Cisco's security technologies.

The Written Test

Cisco's written exam (350-018) for CCIE Security covers the following areas of knowledge:

- Security protocols
- Operating systems
- Application protocols
- General networking
- Security technologies
- Cisco security applications
- General security knowledge
- General Cisco IOS knowledge

NOTE

A detailed blueprint of the CCIE Security written exam is available on Cisco's Web site at www.cisco.com/go/ccie.

The written exam is a computerized multiple-choice test and contains 100 questions. The candidate is allotted two hours to complete the test to demonstrate comprehensive knowledge in each of these areas in order to pass the written exam and qualify to take the lab exam.

The Lab Exam

Where the written exam is of a more theoretical, "book knowledge" nature, the CCIE Security lab exam validates actual hands-on skills in building and troubleshooting an internetwork built with Cisco technologies. The CCIE Security lab exam requires a solid understanding of routing and switching, augmented by firewall and VPN knowledge.

It should be noted that achieving CCIE certification depends on the candidate's preparation as a combination of self-study, training, and work experience. It is unlikely that training or self-study alone will be enough to pass the CCIE exam,

since in-depth knowledge of Cisco commands and architecture is required. The candidate should be very familiar with the following equipment and services:

- 2500 series routers
- 2600 series routers
- 3600 series routers
- 4000 and 4500 series routers
- 3900 series token ring switches
- Catalyst 5000 series switches
- PIX firewalls
- Certificate Authority Support
- Cisco Secure Access Control System
- Cisco Secure Intrusion Detection System

CSPFA: The Exam

The Cisco Secure PIX Firewall Advanced exam (9E0-111) is one of the four exams required for CSS-1 certification and is the focus of this book. This computer-based exam, 75 minutes in duration, includes 55 to 65 questions. This book covers all the objectives of the CSPFA exam and in most cases overshoots them. The goal of this book is not only to provide the knowledge needed to pass the CSPFA exam but also to provide real-world insights that will help you better deploy and manage Cisco PIX firewalls in your environment.

Exam Objectives

The CSPFA exam covers the following topic areas:

- Cisco PIX Firewall technology and features
  - Firewalls
  - PIX Firewall overview
- Cisco PIX Firewall Family
  - PIX Firewall models
  - PIX Firewall licensing
- Getting started with the Cisco PIX Firewall

- User interface
- Configuring the PIX Firewall
- Examining the PIX Firewall status
- Time setting and NTP support
- ASA security levels
- Basic PIX Firewall configuration
- Syslog configuration
- Routing configuration
- DHCP server configuration

- Translations and connections
  - Transport protocols
  - Network Address Translation
  - Port Address Translations
  - Configuring DNS support
- Access control lists and content filtering
  - ACLs
  - Using ACLs
  - URL filtering
- Object grouping
  - Overview of object grouping
  - Getting started with group objects
  - Configuring group objects
  - Nested object groups
- Advanced protocol handling
  - Advanced protocols
  - Multimedia support
- Attack guards, intrusion detection, and shunning
  - Attack guards
  - Intrusion detection

- Authentication, authorization, and accounting
  - Introduction
  - Installation of CSACS for Windows NT
  - Authentication configuration
  - Downloadable ACLs
- Failover
  - Understanding failover
  - Failover configuration
  - LAN-based failover configuration
- Virtual private networks
  - P:IX Firewall enables a secure VPN
  - IPsec configuration tasks
  - Prepare to configure VPN support
  - Configure IKE parameters
  - Configure IPsec parameters
  - Test and verify VPN configuration
  - Cisco VPN client
  - Scale PIX Firewall VPNs
  - PPPoE and the PIX Firewall
- System maintenance
  - Remote access
  - Command-level authorization
- Cisco PIX Device Manager
  - PDM overview
  - PDM operating requirements
  - Prepare for PDM
  - Using PDM to configure the PIX Firewall
  - Using PDM to create a site-to-site VPN
  - Using PDM to create a remote access VPN

Summary

In this chapter, we learned about the importance of security to any organization deploying networks today. Threats can come from both outside and inside. A security strategy must address issues of confidentiality, integrity, availability, authentication, access control, and auditability.

Every organization with an IT infrastructure needs an information security policy. The policy development and maintenance process should include multiple stakeholders representing the different areas of the organization, and it must take into account the overall risk picture.

Cisco's Security Wheel describes an ongoing process of securing your network, monitoring and responding to incidents, testing for vulnerabilities, and managing and improving security.

Firewalls are devices that regulate and filter traffic between networks. The most common deployment is on an Internet connection, but more and more organizations are using firewalls internally to segment sensitive areas. There are two fundamental approaches to firewall design: packet filtering, which operates at the network layer, and application proxying, which works at the application layer and understands details of particular applications. Packet filters have the advantage of speed, but proxies have the advantage in security. Stateful packet filters, an evolution of basic packet filters, have the intelligence to keep track of connections to make more informed pass/block decisions.

Firewall architectures often include one or more DMZ networks, which enable services to be made available to the Internet while keeping them protected by the firewall and segmented from the internal LAN.

Network Address Translation allows an organization to use private, non-unique addresses on their internal networks. These addresses are translated to globally unique addresses for routing on the Internet. NAT also provides security by hiding internal network details from the outside.

Virtual private networks are supported by most major firewalls today. They enable remote sites and users to gain authenticated, confidential access to the enterprise from the Internet.

Cisco offers two security-specific certification programs: CSS-1 and CCIE Security. CSS-1 requires the CCNA certification and passing of four written tests that cover security fundamentals, VPNs, PIX firewalls, and intrusion detection. CCIE Security is a more advanced certification and requires a rigorous hands-on lab exam in addition to a difficult written exam.

Solutions Fast Track

The Importance of Security

- ☑ Information security is more important than ever due to the interconnectedness of businesses and the increased sophistication of hackers.
- ☑ Fundamental areas of security include confidentiality, integrity, availability, authentication, authorization, and auditability.
- ☑ The Internet and its associated protocols were not initially designed to be secure. This means that extra effort is required to secure information assets using defined and documented processes, additional technologies, and security awareness.
- ☑ The greater threat to an organization comes from employee and contractor misuse on the inside. Perimeter defense is important but should not be the only area of effort.

Creating a Security Policy

- ☑ A good security policy forms the foundation for all other information security activities. It should be general in scope so that changes in people or technology do not require that the policy be changed as well.
- ☑ Participation from key stakeholders in the policy development process is essential to gaining support for the policy.
- ☑ The policy process should include a companywide risk assessment and documentation of the critical information flows.
- ☑ The high-level policies flow down and guide creation of specific standards, processes, and procedures.

Cisco's Security Wheel

- ☑ The Cisco Security Wheel is a model that graphically represents the ongoing process nature of security.

- ☑ Based on the security policy, the Wheel includes four major functions: secure, monitor and respond, test, and manage and improve.
- ☑ Many tools, both commercial and free, are available to support each function in the Security Wheel.

Firewall Concepts

- ☑ Firewalls are most often placed between an organization's internal network and the Internet, although they are increasingly used within the internal LAN to separate different zones of trust.
- ☑ There are two fundamental approaches to firewall design: packet filters and application proxies. Many packet filters offer the ability to keep track of active connections (*statefulness*) and in general offer much faster performance and the most flexibility. Application proxies are considered more secure but require that a proxy agent be available for each application running through the firewall.
- ☑ Firewall policies should be assiduously documented with business justification, with a defined process for making changes.
- ☑ Address translation allows use of private, nonroutable IP addresses on the internal (local) network, which are translated at the firewall into globally unique addresses for routing on the Internet.
- ☑ Most firewalls support virtual private networking (VPN) capability, which allows other sites and remote users to connect to the enterprise network through encrypted tunnels.

Cisco Security Certifications

- ☑ To achieve the Cisco Security Specialist 1 certification, you need to demonstrate a solid understanding of Cisco network security, PIX firewalls, VPN solutions, and Cisco Secure IDS by taking four written exams. CCNA certification is a prerequisite.
- ☑ CCIE Security is extremely complex and requires detailed knowledge of networking, PIX firewalls, and VPNs. The CCIE Security process includes both a written exam and a hands-on lab exam.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: How do I convince my managers of the need for security and get more funding?

A: Unfortunately, managers in many organizations have not expanded their definition of business risk to include risk to information assets. The problem is that generally, most other risks are quantifiable, and it is a straightforward calculation to determine how much money should be spent to mitigate those risks, if any. Information security is a thornier problem in that hard-and-fast numbers are not available to enable an organization to determine how likely it is that they will experience a security incident and how much it will cost. It is becoming easier to calculate these numbers based on various industry surveys and direct loss experiences, but the seemingly random nature of attacks makes such quantification tough.

Management often views information security as spending money (often lots of it) to protect against something that might never happen. It frequently takes an actual serious breach or worm infestation to "shake the money tree." In the (fortunate) absence of that event, you should collect as much data as you can. Participate in trade groups and information security associations so you can talk to others in your industry or field. Document carefully the risks and threats you face, along with descriptions of the business benefits that the spending will result in. The need for security is real, and you must convince your management of that.

Q: How can I get a policy developed when my company takes a very casual and trusting approach to security?

A: Talk to the various stakeholders in your company about what they perceive as the key risks. Every company has risks, and the company culture does not change that. Try to convince the stakeholders of the benefits of protecting information assets—if not from employees, at least from outside attackers. Creating an acceptable use policy is a great start.

Q: I do not have enough staff to adequately manage security. How can I keep on top of everything?

A: You need to prioritize your activities and automate wherever possible. Perform a risk analysis, evaluate where the greatest threats are, and do what is necessary to protect against them. Build a secure baseline configuration for all your OS platforms from which all new systems are built. Develop a good configuration management process to make it easier to stay current on patches. By making a strong initial effort to secure your network, you will experience less tactical firefighting.

Q: I have a new Web application that needs to communicate with a database server on my internal LAN. How do I make this application secure with my firewall?

A: Place your Web server on the DMZ network. Create rules to filter traffic from the outside coming into your Web server. Accessible ports should be only HTTP (TCP 80) and HTTPS (TCP 443) and any others necessary for the application to run. Then restrict inbound traffic to come from the Web server IP address only, going only to the database server IP and destination port number(s). Monitor this backend connection continuously, and deploy network-based intrusion detection on the DMZ as well as host-based intrusion detection on the Web and database servers to detect malicious activity.

Chapter 2

Introduction to PIX Firewalls

Solutions in this chapter:

- PIX Firewall Features
- PIX Hardware
- PIX Licensing and Upgrades
- The Command-Line Interface

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

Good security administration is labor-intensive, and therefore organizations often find it difficult to maintain the security of a large number of internal machines. To protect their machines from outside subversion, organizations often erect a security wall, or "perimeter." Machines inside the perimeter communicate with the rest of the enterprise (or the Internet) only through a small set of carefully managed machines called *firewalls*. These devices allow for access controls that might not be native to the protected hosts; in addition, they can provide authorization or audit controls at the network layer.

Increasingly, these firewalls provide additional security or performance services; since they sit at a point in the network that mediates all communication with the end host, various kinds of service extensions can naturally be integrated into them.

Even in high-security environments, where the resources to harden and provide ongoing security support for the end application are available, firewalls can play an important role. In addition to the features described previously, firewalls can support the concept of defense in depth: Multiple protective technologies support higher levels of trust in case of error or omission at one layer. Having multiple controls also supports the concept of separation of duties: Different groups can support application layer and network layer securities, ensuring that no single person or group can compromise the system. Firewalls are thus an essential part of every network security design.

Cisco's PIX firewalls are a series of appliances that offer world-class security and high levels of performance and reliability. They are a mature product, having been a part of enterprise and service provider networks since 1995. Cisco PIX firewalls fit into a wide range of environments, from small office/home office (SOHO) environments to large enterprises and service providers. With support for complex protocols, the latest VPN technologies, and intrusion detection features, the PIX is one of the leading firewalls in the market.

In this chapter, you will learn about some of the main features that Cisco PIX firewalls have to offer. We will look at the different models of PIX and the types of environment in which they fit. We will then perform basic configuration on a PIX firewall through the command-line interface.

PIX Firewall Features

The PIX 500 series firewalls are a market-leading security appliance, and for good reason. They provide robust performance in a firewall while providing a

highly scalable architecture ranging from plug-and-play SOHO devices to carrier-class firewalls with gigabit connections. They provide protective services that define what a firewall should do. From stateful packet inspection to content filtering, VPN termination to address translation, support for PKI applications, and providing security to multimedia applications, the PIX does it all.

With such flexibility comes the requirement to configure the devices correctly. Luckily, for those who are already comfortable with a router prompt, the PIX is based on a familiar command prompt. Of course, the PIX fits into standard Cisco management tools such as CiscoWorks, so it will seamlessly integrate into your LAN/WAN environment.

Embedded Operating System

Many firewalls are based on general-purpose operating systems. This means that maintenance is required to ensure not only correct configuration but that the base operating system is patched and secured. This requirement offers both a higher long-term cost as well as the potential for security weaknesses.

An embedded operating system is one in which the OS is self-contained in the device and resident in ROM. This involves reduced maintenance costs, since no customizations or OS configurations are required; a single image is downloaded and stored to flash. It means that there is little that can go wrong; you cannot accidentally leave an unnecessary service running, since the firewall has all its services tuned to only those features appropriate for a security device.

Unlike some appliances that are based on a general kernel such as Linux or Windows CE, the PIX is based on a hardened, specialized OS specific to security services. This OS allows for kernel simplification, which supports explicit certification and validation: The PIX OS has been tested for vendor certification such as ICSA Labs' firewall product certification criteria as well as the very difficult-to-obtain International Standards Organization (ISO) Common Criteria EAL4 certification. This testing allows for maximum assurance in deployment from Cisco's positive security engineering based on good commercial development practices. Kernel simplification has advantages in throughput as well; the PIX 535 will support up to 256,000 simultaneous connections, far exceeding the capabilities of a UNIX- or Windows-based OS on equivalent hardware.

One key advantage to the software on a PIX firewall is its similarity to Cisco IOS. This means that internetworkers have the ability to rapidly master management of the PIX, reducing deployment costs and supporting management by network operations center (NOC) personnel. You should not have to be an expert in UNIX or Windows 2000 to be able to deploy a VPN or firewall!

The Adaptive Security Algorithm

The heart of the PIX is the Adaptive Security Algorithm, or ASA. The ASA is a mechanism to determine if packets should be passed through the firewall, consistent with the information flow control policy as implemented in the access control list (ACL) table. The PIX evaluates packet information against developed state and decides whether or not to pass the packet.

Let's go through this process one step at a time. First there is the concept of a *datastream*. Packets that are flowing across a wire have identifying characteristics: IP address of source and destination, sometimes numbers associated with the type of communication (ports) of source and destination, and numbers such as IP identifiers or synchronization and acknowledgement numbers that identify where a packet belongs in a particular connection. When you open a Web page—say, to www.cisco.com/index.html—you establish a connection between your browser and the Web server. One piece of HTML is transferred; if it has not been cached, this page represents about 90K of text. That text may then open up additional connections for all the embedded pictures. The process involves a "dance" between browser and server—a "handshake" to initialize the connection, a "get" to specify the data being requested, a "response" to say if the data is available, and the actual data itself. Since the file is so large, these steps all occur in multiple packets between browser and Web server, with data flowing down from the server and acknowledgment of receipt of data flowing up from the browser.

The information flow control policy is an expression of the information that is allowed to flow through the network. A sample policy might be, "If the datastream was initiated by someone on the inside, let it pass; if the datastream was initiated by someone from the outside, block it."

An ACL table is a mechanism via which you can try to implement this policy. It compares those distinguishing numbers against a database to see if the packet is consistent with policy. If it is not allowed by the database, the packet is dropped and perhaps logged.

The earliest routers used fixed-access control lists to determine if a packet should be routed; they compared fundamental information about the packet, such as the IP address of the source or destination or the type of service requested or, for some services such as TCP, individual flags on the packets. Then, based on fixed rules, they decided to route the traffic or to drop it. For example, the fixed rules might allow any packet that might possibly be a "return" packet, since under certain circumstances such a packet would be valid. This isn't too much of a problem, since a "return" packet, if it hasn't been requested by the original host,

should be dropped by the host. However, that can cause some information to leak out, so it is helpful to get rid of such packets if we can.

The concept of *state* is the idea that ACLs should probably change over time. A stateful packet filter allows for dynamic rule bases—for example, if the packet is coming from the outside toward the inside, you should check to see if this packet was part of a previously opened datastream. Now, we only allow packets back in if they were previously authorized; that Cisco Web server can't decide to send us data unless we previously requested it.

The biggest problem with fixed rules is that in order to allow certain kinds of traffic—FTP, for example—overly permissive ACLs would need to be implemented. In FTP, two TCP data flows are developed. One, the command channel, runs from the client out to the user—from the inside to the outside. Routers would generally be able to determine the direction of this flow and allow that traffic, as described previously. The second, the data channel, is negotiated by the FTP server and flows from the server back into the client—from the outside to the inside. Moreover, the TCP port—a service identifier telling you an identifier for the port—varies depending on how many files the server has transferred since reboot; thus the ACL would have to allow *all* inbound traffic in a wide range of TCP ports. This means that a malicious user would have free run of the network in those ranges. So router ACL-based firewalls are little more than Swiss cheese enforcement points!

The smart idea is to watch for the negotiation between the FTP server and client. That's part of the concept of state. Armed with that piece of information, the firewall can open only the necessary port for the inbound data flow, and open it only while the transfer is active—dynamically changing the ACLs over time. This allows the firewall to permit authorized traffic and disallow inappropriate traffic with far more sophistication than a static rule.

State

More deeply, *state* is a way of saying that the firewall is maintaining a history of the traffic that has passed and will compare the new packet against previous history to see if the packet is allowed by the information flow control policy rules. There is also a performance benefit of maintaining state: If a packet can be determined to be similar to those already passed, a full analysis against the firewall policy rules does not need to be followed, it can be passed based on the existing state. This allows the PIX to perform at line rate where static access lists might bog down.

One key piece of state is to record active connections. If we can add something to a connection table when it first starts and remove that thing from a connection table when the connection is (gracefully) closed, we have a leg up for that concept of "similar to those already passed." This data is stored in the connections table (CONN).

The PIX has the ability to rewrite the characteristic information described previously, such as IP address and port data. Thus another piece of state is to remember what IP address and port data the PIX has seen lately as well as remembering what it did with them before. It needs to remember how it translated something from a protected net into the outside world. This data is stored in the translations table (XLATE).

Here are the XLATE and CONN tables' output as displayed by PIXOS on a quiet firewall:

```
PIX1# show xlate
3 in use, 112 most used
PAT Global 63.110.38.230(1225) Local 10.10.10.11(32775)
PAT Global 63.110.38.230(22451) Local 10.10.10.11(4025)
PAT Global 63.110.38.230(22450) Local 10.10.10.11(32778)
PIX1# show conn
1 in use, 26 most used
TCP out 63.122.40.140:21 in 10.10.10.11:32775 idle 0:00:10 Bytes 154
    flags UIO
```

This code shows that someone on machine 10.10.10.11 has connected to 63.122.40.140 on port 21 (FTP). The translation maps between socket 63.110.38.230, 1225 on the outside and socket 10.10.10.11, 32775 on the inside. The flags from the connection table are showing that the connection is up and that there is inbound and outbound data. A little while later:

```
PIX1# show conn
1 in use, 26 most used
TCP out 63.122.40.140:21 in 10.10.10.11:32775 idle 0:06:48 Bytes 216
    flags UFRIO
```

Notice that the idle counter is larger (the traffic flow has been idle, no packets have been received), a few more bytes have passed, and the flags now have *F*, for *outside FIN*, and *R*, for *outside acknowledged FIN*.

This indicates that the firewall has taken notice of the transfer. In addition to the basic housekeeping of passing traffic appropriately (there is address translation

going on, so that must be addressed), the PIX is keeping an eye on the transported traffic. Port 21 is FTP, so it knows that there might be an inbound connection. It knows from the first output that traffic between those two machines on those socket pairs is expected and should be passed. It knows from the second output that traffic between those two machines should no longer occur, because the sides have reset each other, and that any stray packets are now either lost retransmissions or someone doing something they should not. The firewall has "learned" about the transfer over time and is able to change its rules in response to past traffic.

Security Levels

When firewalls were first implemented, they typically had only two interfaces: the outside, or "black," network and the inside, or "red," network. These interfaces corresponded to degrees of trust: Because the inside was controlled and was "us," we could allow pretty much anything originating in the red network to travel to the black network. Furthermore, because the outside was "them," we limited pretty much anything originating in the black network to come inside the firewall.

The modern style is to have a DMZ, or multiple service networks. This makes the idea of "us vs. them" much more complex. The PIX 535 has a modular chassis with support for up to 10 interfaces! Using the *nameif* command, you can assign a security level, an integer between 0 and 100. Make sure that each interface has a different value. When you are designing your security zones, the idea should be to order the zones by degrees of trust and then assign integers to the levels, corresponding to how much you trust the network—0 for the outside (untrusted network), 100 for the inside (trusted network), and values between 0 and 100 for relative trust.

How ASA Works

Informally, ASA allows traffic to flow from a higher security level to a lower security level, unless modified by the *conduit* or *access-list* commands. More formally, the manual notes:

- No packets can traverse the PIX firewall without a connection and state.
- Outbound connections or states are allowed, except those specifically denied by access control lists. An outbound connection is one in which the originator or client is on a higher security interface than the receiver or server. The highest security interface is always the inside interface and the lowest is the outside interface. Any perimeter interfaces can have security levels between the inside and outside values.

- Inbound connections or states, except those specifically allowed, are denied. An inbound connection or state is one in which the originator or client is on a lower security interface or network than the receiver or server. You can apply multiple exceptions to a single *xlate* (translation). This lets you permit access from an arbitrary machine, network, or any host on the Internet to the host defined by the *xlate*.
- All ICMP packets are denied unless specifically permitted.
- All attempts to circumvent the previous rules are dropped and a message is generated. It is sent to a management device (local buffer, SNMP trap, syslog, console), depending on the severity of the attempt and local configuration. (Note that normal traffic might also trigger logging, again depending on configuration. At the highest debugging mode, every packet generates an alert!)

Technical Details for ASA

The PIX is an Internet Protocol firewall. It accepts and passes only IP packets; all others are dropped. It is worth taking a moment to look at the details of the protocols to see what the PIX is looking at and how it uses that information.

Internet Protocol

IP is an unreliable, routable packet delivery protocol. All upper-layer protocols use IP to send and receive packets. IP receives segments from the transport layer, fragments them into packets, and passes them to the network layer.

The IP address is a logical address assigned to each node on a TCP/IP network. IP addressing is designed to allow routing of packets across internetworks. Since IP addresses are easy to change or spoof, they should not be relied on to provide identification in untrusted environments. As shown in Figure 2.1, the source and destination addresses are included in the IP header.

Let's quickly review the meaning of key fields in Figure 2.1. Most are not specifically part of the review exam, but it helps to put what the PIX does in context:

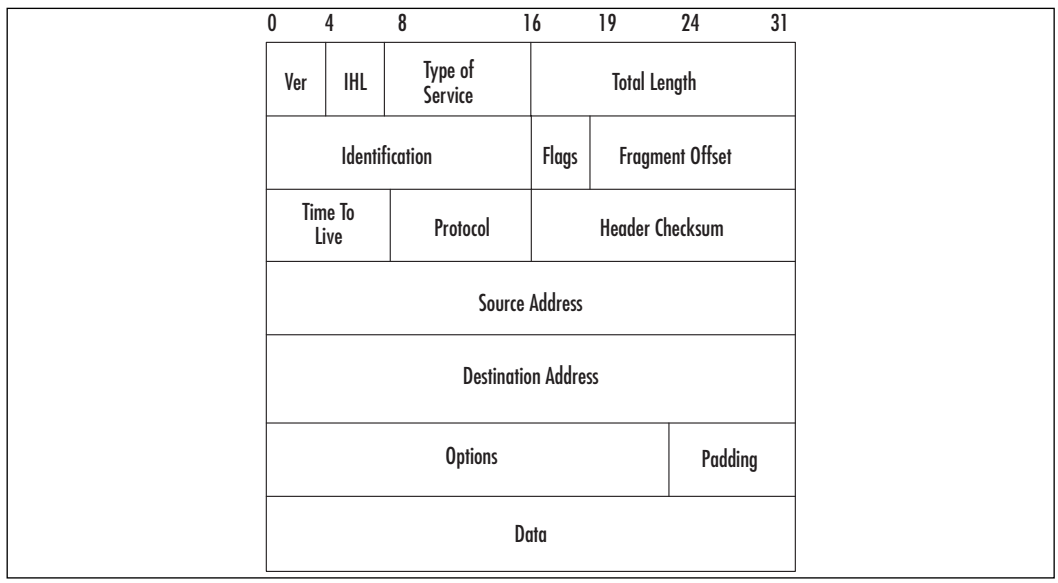
- The *protocol* parameter indicates the upper-level protocol that is using IP. The decimal value for TCP is 6, and UDP is 17. The list of assigned numbers for this field is available at www.isi.edu/in-notes/iana/assignments/protocol-numbers. Note that this field is important for *access-list* commands. The command syntax is:

```
access-list <acl_ID> {deny | permit} <protocol>…
```

The protocol number here corresponds to this field. Note that you can specify the keyword *tcp* for type 6 or *udp* for type 17.

- The *source address* and *destination address* fields are filled with the IP addresses of the respective devices; note that an IP address is four octets, so this can be viewed as a 32-bit number. You will see these numbers in the XLATE table.

Figure 2.1 The IP Header



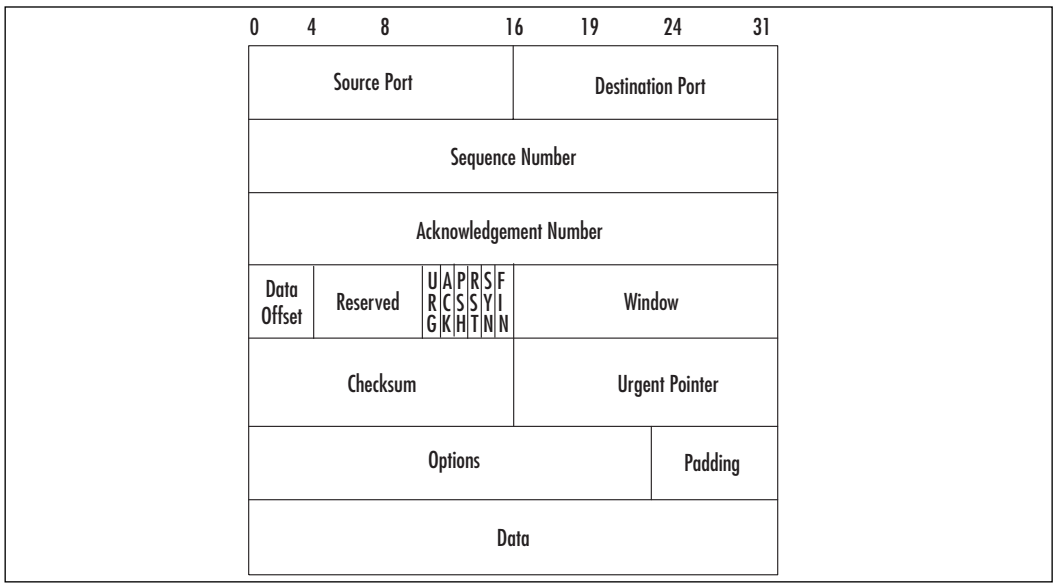
Transmission Control Protocol

Many Internet services, such as HTTP, SMTP, or ssh, are based on TCP. This protocol provides reliable service by being connection-oriented and includes error detection and correction. The connection must be established before a data transfer can occur, and transfers are acknowledged throughout the process. Firewalls can identify the connection establishment and often interrupt that establishment as part of the protective mechanism. Acknowledgments assure that data is being received properly. The acknowledgment process provides robustness in the face of network congestion or communication unreliability. The acknowledgment has also been used to penetrate stateless firewalls; the PIX can identify packets that are not part of valid streams and block transmission. TCP also determines when the transfer ends and closes the connection, thus freeing resources on

the systems. As noted earlier, the PIX watches for transfer end and acts appropriately. Checksums assure that the data has not been accidentally modified during transit. The PIX has the ability to rewrite checksums to handle NAT issues.

Figure 2.2 shows the format of the TCP header.

Figure 2.2 The TCP Header



The PIX inspects TCP packets for several fields, notably source port, destination port, sequence and acknowledgment numbers, and TCP flags. Notice that source and destination ports and information about the flags are listed in the CONN connections table.

The concept of *port* is common to both TCP and UDP (discussed in the following section). The idea is that for these types of protocols, we can identify an ordered pair (IP address and port), called a *socket*, with each side of the communication flow. Multiple communications from the same host (same IP) can be distinguished by different port numbers—thus different sockets.

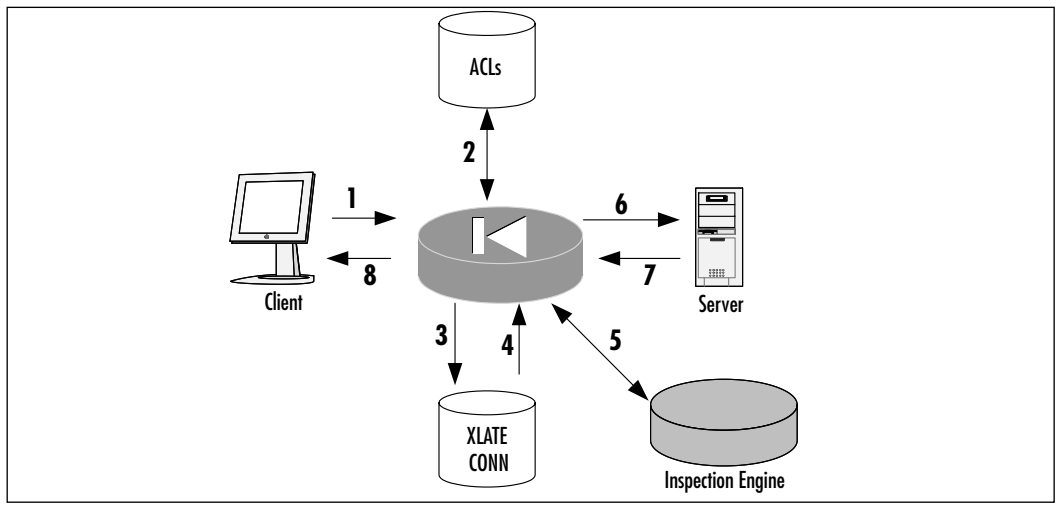
Sockets on the server generally have a "well-known port" number. The PIX has a mapping between well-known ports and their English equivalents.

We have enough background to see how ASA works for TCP connections. A TCP datastream begins with the "three-way handshake." The idea is for each side to set up the *initial sequence number*, a pointer that will describe the position in the datastream for each packet sent. The TCP flag that indicates a request to start that datastream is the SYN flag. So the first three packets are an initial SYN request

from the client to the server; then back from the server to the client with acknowledgment of the client's request (by setting the ACK flag) and the server's need to initialize as well (by setting the SYN flag); and finally the client back to the server, acknowledging the server's synchronization request. So, from the TCP level, the path is SYN, SYN/ACK, ACK.

At the PIX, a little more goes on. Figure 2.3 provides a diagram for how information flows through the PIX. Let's follow the first two network packets.

Figure 2.3 Basic ASA Operations



1. The client generates a SYN packet, headed toward the server, to establish a new connection.
2. The PIX investigates the ACL to determine if the information flow control policy should permit the new connection.
3. Assuming the connection is valid, the PIX updates the connections table.
4. The XLATE table is updated as necessary.
5. The stream is processed by the Application Inspection Engine, if necessary, which could involve rewriting the packet.
6. The packet is sent on to the server.
7. On the reverse path, the server responds with its SYN/ACK.
8. However, since this is not an initialization request, inspection of the rule base is not required; it looks the packet up in the connections table and then forwards it back to the client.

Designing & Planning…

TCP Sequence Number Randomization

All that SYN and SYN/ACK work is designed so that both sides will agree on an initial sequence number (ISN) for each side of their communication. This adds a layer of security protection; in theory, one would have to be able to "hear" the TCP SYN request to know what ISN to use, and thus the IP address of the host in the datastream must be able to receive the packet, and therefore, for example, hosts on the Internet can't masquerade as local hosts.

Unfortunately, many servers use an easily guessed ISN generation function. One famous break-in, Kevin Mitnick's raid on Tsunomo Shinomura's data, chronicled in the book *Takedown*, was based on this flaw. The PIX provides protection against this sort of attack by using TCP sequence number randomization. As the packets pass through the firewall, they are rewritten so that the ISNs cannot be predicted.

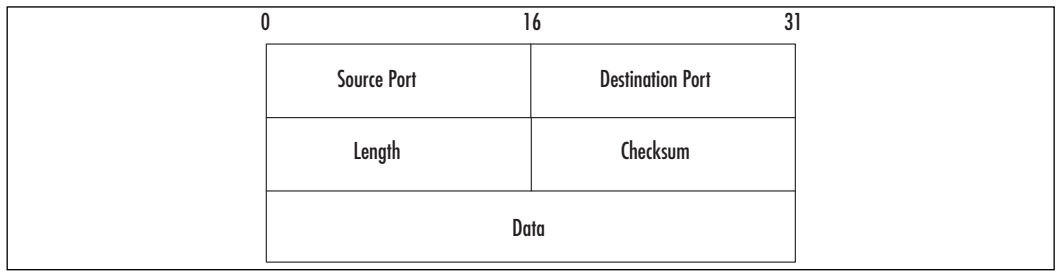
This system is not perfect; you should still use authentication and authorization at the server where available. But it should provide an extra layer of protection that will let your security officers sleep better at night.

User Datagram Protocol

Several Internet applications, notably Domain Name Service (DNS) and many streaming audio and video protocols, are based on User Datagram Protocol (UDP). The UDP protocol is a simple, unreliable transport service. It is connectionless, so delivery is not assured. Look at the simple design of the UDP header in Figure 2.4 and you will understand this protocol's efficiency. Since connections aren't set up and torn down, there is very little overhead. Lost, damaged, or out-of-order segments will not be retransmitted unless the application layer requests it. UDP is used for fast, simple messages sent from one host to another. Due to its simplicity, UDP packets are more easily spoofed than TCP packets. If reliable or ordered delivery of data is needed, applications should use TCP.

There is usually a trade-off between simplicity and security, and this is true with UDP. Because TCP is connection oriented, we can identify the start of the session by unique flags—but as you can see in Figure 2.4, there aren't any flags here. All you have to work with is the UDP socket pairs.

Figure 2.4 The UDP Header



This is where the firewall state comes in. The PIX has the ability to recognize the first UDP packet in a datastream. When the first packet is permitted by the information flow control policy (either because it is coming from a trusted net toward a less trusted one or because of an explicit exception in the ACL), the same sort of process shown in Figure 2.3 occurs. If permitted, an entry is made in the connections table, and further packets with the same socket pairs are associated with that authorized datastream until an idle timeout occurs. (The idle timeout is set with the *timeout* command and defaults to 2 minutes.)

Note that other protocols besides TCP and UDP are permitted. Most common is ICMP, the Internet Control Message Protocol. ICMP provides diagnostic functions and error reporting for IP. For example, ICMP can provide feedback to a sending host when a destination is unreachable or time is exceeded (TTL=0). A ping is an ICMP echo request message, and the response is an ICMP echo reply.

Other types of protocols are filtered by the PIX, although the concept of socket does not apply (and so you cannot specify extra parameters on the access list beyond filtering on the source and destination addresses). The special protocol 0 refers to any IP packet, and you can specify any value between 0 and 255. You can also use literals; you have already seen the literals TCP (which is 17), UDP (which is 6), and ICMP (which is 1).

These other protocols are handled similarly to the UDP approach, with idle timeouts removing entries from the connection table when they are no longer valid.

Advanced Protocol Handling

The PIX has taken elements from both camps in an example of a hybrid firewall, combining stateful packet filtering with advanced protocol handling with proxies via the *fixup* command. For common applications, the PIX provides advanced protocol handling, not only dealing with embedded IP addresses (the scourge of NAT functionality) but improving overall security handling.

Providing support for complex protocols is a distinguishing characteristic of the PIX. The "fixup" proxies include ftp, http, h323, ils, rsh, rtsp, smtp, sip, skinny, and SQL. Some protocols, such as DNS Guard (which prevents multiple DNS responses from penetrating to the host), are supported in the native PIX services and do not need to be configured.

Application support of this type is where the real power of a firewall shines. The PIX is more than just a gatekeeper, passing or blocking packets; it understands the underlying protocol and actively rewrites the communications—enforcing RFCs, eliminating dangerous commands, and preventing the leakage of information—to provide the highest level of security available, consistent with application functionality.

VPN Support

An important aspect of network security is confidentiality of information. Packets flowing along a network are much like postcards sent through the mail; if you don't want the world reading your messages, you have to take additional care.

To achieve the kind of confidentiality offered on a private network, several approaches have been followed. One is to use encryption to conceal the information. An early standard, followed by Microsoft, is the Point-to-Point Tunneling Protocol, or PPTP. Much like putting a letter inside a sealed envelope, this standard allows encapsulating (and concealing) network traffic inside a transport header. A similar but more comprehensive approach is to use the Layer 2 Tunneling Protocol, or L2TP. This protocol is native to many Microsoft deployments, and so the PIX's support for PPTP and L2TP is an important element of the feature set.

In the fall of 1998, the Security Architecture for IP (IPsec) was published in RFC 2401. Cisco has provided a leadership position in IPsec implementation, having co-authored many of the IPsec RFCs as well as providing solutions for some of the stickier IPsec issues, such as NAT traversal. It should be no surprise that the PIX is an excellent IPsec tunnel terminator. It has a wide range of interoperable standards and is straightforward to configure with pre-shared keys or with a certificate authority. Many companies are using the PIX as an integrated firewall/VPN terminator, particularly in SOHO environments, as well as a standalone VPN terminator in conjunction with another (dedicated) firewall. Details on VPN configuration are provided in Chapter 7.

One of the PIX's best features is VPN performance. The models are designed to produce essentially wire-speed performance under heavy IPsec load. Because of the simplicity of the appliance's maintenance, VPN termination on a PIX is a sound choice for many enterprise or carrier-class environments.

URL Filtering

A uniform resource locator, or URL, is the way we identify addresses for information on the World Wide Web (WWW). The PIX firewall supports URL filtering by capturing a request and querying a database located on an N2H2 or Websense server. The N2H2 server can be running Linux (see www.n2h2.com/products/bess.php?os=lnx&device=pix) or Microsoft Windows (see www.n2h2.com/products/bess.php?os=win&device=pix); the Websense server can use these platforms or be installed on a Solaris server (www.websense.com/products/integrations/ciscoPIX.cfm).

URL filtering provides you with a way to apply an acceptable use policy for Internet browsing as well as to capture and analyze how your personnel are using the Internet. The servers themselves provide reporting capabilities so that you can determine how well your policy is being followed.

NAT and PAT

Another key strength of the Cisco PIX is its ability to translate addresses. Historically, an insider note is that the PIX comes from equipment created by a company called Network Translations Inc., and the PIX's first role was simply to perform address translation. (The name *PIX* comes from *Private Internet Exchange*, reflecting its purpose: to exchange traffic between private networks and the Internet.)

Network Address Translation, or NAT, encapsulates the idea that we can remap IP addresses (or sockets) where desirable in order to provide efficiencies or security. In the late 1990s, there was a great concern that we would run out of IP addresses; every host needed its own IP, and there are only 2^{32} to go around. Once we hit that number of computers, we'd be out of addresses. Worse, when you changed service providers, you generally had to give up your IP addresses and renumber all your machines—an expensive, time-consuming task that often ended up missing some machines, leaving them unable to communicate.

An idea was developed to use "private" addresses internally and, at the perimeter of our control, remap them into "public" addresses given to us by our service provider. Now we do not have to spend a lot of time renumbering our IP addresses; if we change providers, we only have to change the value of the IP addresses on the external firewalls and we are done. In February 1996, Cisco co-authored RFC 1918, which established ranges for "private" addresses—all of the 10 network (10.0.0.0 through 10.255.255.255), part of the 172 network (172.16.0.0 through 172.31.255.255), and the 192.168 network (192.168.0.0

through 192.168.255.255). This RFC is followed nearly universally by enterprises today, with IP address schemes chosen from these private networks to simplify the structure of the internal network.

NAT also provides a form of "security through obscurity." Since the private addresses are not advertised, an outside attacker does not necessarily know how the machine refers to itself; this structure adds an extra layer of work the attacker needs to perform to understand how to connect to an internal host.

There are several different ways to perform the address translation. The simplest form of NAT provides a one-to-one map between internal host IP addresses and external addresses—for example, a map between 10.1.1.1 and 198.133.219.25. Then any reference, say 198.133.219.25 port 80, gets translated to 10.1.1.1 port 80, and vice versa. This form of NAT has two different flavors: static NAT, in which the translation is set up once and is permanent, and dynamic NAT, in which a translation is set up from a pool of available addresses and is torn down when an idle timeout occurs. The former is perfect for remapping servers that need to provide consistent access to the outside world; because the translated address is fixed, it can be put into public DNSs and readily accessed by outside clients. The latter is perfect for remapping users who need public services and IP addresses for a short time, which can then can be released for other users when the services and addresses are no longer needed. This system allows for, say, 100 people to hide behind 30 addresses, as long as no more than 30 of those people need external access at any one time.

The idea of dynamic NAT can be extended even further. Most IP services are based on sockets, such as IP address/port number pairs. Rather than remapping on IP address, we can remap on sockets. Now 10.1.1.1,80 might get mapped to 198.133.219.25,3125 while 10.1.3.42,80 gets mapped to 198.133.219.25,4176—the same IP address in both cases, but because the port numbers are different, the sockets are different. Therefore, the other side of the conversation would be able to distinguish between these two datastreams.

This concept is called *Port Address Translation (PAT)* and allows for stacking over 30,000 TCP sessions on a single IP address. The good news is that now when you want to hide your 100 users, you can hide them behind a single IP address. The bad news is that certain protocols—ones that expect fixed port addresses—are broken by this translation. The PIX can be configured to use static addresses for fixed servers and dynamic addresses for users with an overflow pool of PAT (or even multiple PAT to give a better chance of being able to preserve port address). You can see that the PIX is a very flexible and highly effective network address translation device.

High Availability

The three fundamental concepts of information security are confidentiality, integrity, and availability. The PIX addresses the availability idea by providing a robust, fault-tolerant environment. *Fault-tolerant* means that if something goes wrong, alarms are set off and something is done to ameliorate the problem.

The term *high availability* usually refers to hardware fault tolerance. Obviously, a firewall is a critical piece of equipment: By its very nature, it has to stand in the center of the traffic flow. Cisco hardware is of very high quality, and the PIX has no moving parts, but sometimes equipment does fail. High availability is a device configuration so that isolated failure of the hardware will not bring down your network.

To achieve this goal, of course you must have multiple pieces of hardware. In this case, two PIXs are configured similarly, and they communicate between each other. If one piece of hardware dies, the other transparently picks up the traffic and alarm messages are sent to the network management console.

High availability can be configured in several ways. Naturally, you need a second PIX that will be configured in a hot standby fashion. The simplest and least expensive way is through a serial cable, provided when you purchase the failover license. Alternately, a LAN interface can be dedicated to the failover process. With the failover cable, hello packets containing the number of bytes seen by the interfaces are transmitted between the two boxes, and if the values differ, failover can occur. With the LAN interface, full state information is transmitted so that in the event of a failover, the TCP sessions can keep running without reinitialization.

PIX Hardware

The PIX has many different configuration models to ensure that the product will be suited to different environments. Obviously, the requirements of a SOHO user will be different from those of a service provider. Cisco has provided various classes with different price points to ensure optimum product placement.

Models

Five models are currently supported: the 501, the 506E, the 515E, the 525, and the 535. However, there are three models that you might see deployed in enterprise environments: the 506, the 515, and the 520. At a glance, Table 2.1 shows the vital characteristics of each of the models:

Table 2.1 PIX Model Characteristics

Model	End of Life?	Processor Type	Maximum Interfaces	Failover Support	Clear-Text Through-put	VAC Available?	3DES Through-put	RAM Memory
501	No	133MHz AMD SC520	2	No	10Mbps	No	3Mbps	16Mb
506	Yes	200MHz Intel Pentium MMX	2	No	20Mbps	No	10Mbps	32Mb
506E	No	300MHz Intel Celeron	2	No	20Mbps	No	16Mbps	32Mb
515	Yes	200MHz Intel Pentium MMX	6**	Yes	146Mbps	No	10Mbps	64Mb**
515E	No	443MHz Intel Celeron	6**	Yes	188Mbps	Yes	63Mbps*	64Mb**
520	Yes	233MHz Intel Pentium MMX	6	Yes	170Mbps	Yes	60Mbps*	128Mb
525	No	600MHz Intel Pentium III	8	Yes	360Mbps	Yes	70Mbps*	256Mb**
535	No	1GHz Intel Pentium III	10	Yes	1Gbps	Yes	100Mbps*	1Gb**

* Maximum 3DES throughput is achieved with the VAC; ** maximum requires the unrestricted license.

PIX 501

The 501 is the basic entry model for the PIX and has a fixed configuration. It has a four-port 10/100Mbps switch for inside connectivity and a single 10Mbps interface for connecting to the Internet upstream device (such as cable modem or DSL router). It will provide 3Mbps throughput on a 3DES IPsec connection, which should exceed a SOHO user's requirements. The base license is a 10-user license with DES IPsec; optional is a 50-user upgrade and/or 3DES VPN support.

The 501 is based on a 133MHz AMD SC520 processor with 16MB of RAM and 8MB of flash. There is a console port, a half-duplex RJ45 10BaseT port for the outside, and an integrated, autosensing, auto-MDIX 4 port RJ45 10/100 switch for the inside.

PIX 506

The 506 is the basic remote office/branch office device. Once again, the appliance is not hardware configurable, with one console port and two autonegotiate RJ45 10BaseT ports, one for inside and one for outside. Performance is greatly increased; the 506 supports 8Mbps clear-text throughput, with 6Mbps 3DES IPsec, which should permit supporting hundreds of branch office users in a VPN tunnel back to corporate.

The hardware is based on a 200MHz Intel Pentium MMX, with 32MB of RAM and 8MB of flash.

PIX 506E

The 506E product, an enhanced version of the 506, has replaced it on the product sheets. The chassis are similar, but the 506E has a beefier CPU, a quieter fan, and a new power supply. The CPU is the 300MHz Intel Celeron, while the RAM and flash are of the same capacity. Clear-text throughput has been increased to 20Mbps (wire speed) while 3DES throughput increased to 16Mbps. Licensing on the 506E (and 506) is easier than the 501; it is provided in a single, unlimited-user mode. The only extra license you might need is the 3DES license.

PIX 515

The next step up the scale is the PIX 515, intended for the enterprise core of small to medium-sized businesses. Again, this product has wirespeed performance, but this time the pipe is a bit fatter and carries the ability to handle up to 170Mbps of clear-text throughput.

The chassis is a 1U pizza box, intended for rack mounting. Probably the most important difference between the 506 and the 515 is that the chassis is configurable; it comes with a slot for an additional single-port or four-port Fast Ethernet interface, allowing the inside, outside, and up to four additional service networks. The base unit is based on the same 200MHz Intel Pentium MMX with 32MB of RAM and 8MB of flash as the 506E.

The licensing is flexible, so enterprises can purchase only what they need. The restricted license limits the number of interfaces to three and does not support high availability. The unrestricted license allows for an increase in RAM (from 32MB to 64MB) and up to six interfaces, together with failover capability.

PIX 515E

The 515E replaced the 515 in May 2002. It has a higher-performing 433MHz Intel Celeron, increasing base firewall performance. Another new option is the ability to offload the arithmetic load of DES computation from the OS to a dedicated VPN accelerator card (VAC), delivering up to 63Mbps 3DES throughput and 2,000 IPsec tunnels. Licensing is similar: the restricted license limits you to three interfaces and no failover, whereas the unrestricted license has the memory upgrade, the VAC, and up to six interfaces.

PIX 520

The PIX 520 is an odd bird. It was designed as the high-end PIX platform, with the PC-style rack-mount chassis and a wide mix of available media cards, including Token Ring and fiber. Like the earlier PIXs, the 520 comes with a DB9 console port and a diskette drive; it is based on the 200MHz Intel Pentium MMX but with 128MB of RAM. Also unusual is the licensing: Like the 501, the 520's license is based on the number of users. For an entry PIX, you would purchase PIX-CONN-128, which would allow 128 simultaneous users. There were license upgrades to 1024 users or unlimited users.

Having the diskette drive is especially convenient. Although it uses up real estate in the rack, it allows you to have a handy boot medium in case the network goes down or is otherwise inaccessible; TFTP servers are not required. It also allows you to readily reset the password (by booting the appropriate password-clearing binary) or restore to a known good condition. Of course, these features are now achieved through appropriate network management tools, such as CiscoWorks or the PIX Firewall Manager.

PIX 525

The PIX 525 replaced the PIX 520 in June 2001. It is designed for large enterprise or small service provider environments. The diskette drive is gone; however, the 525 still supports single- or four-port 10/100 Fast Ethernet, 4/16 Token Ring, and dual-attached multimode FDDI cards but now also picks up Gigabit Ethernet. Performance tells the story here: Based on the 600MHz Intel Pentium III, the 525 boasts 360Mbps clear-text throughput and, with the accelerator card, 70Mbps of 3DES IPsec tunnel traffic.

Licensing is based on interface counts and failover, as with the earlier models. The restricted license limits the PIX 525 to 128MB of RAM and six interfaces. The unrestricted bumps RAM to 256MB, allows up to eight interfaces, and supports failover. As before, 3DES licensing is separate, if desired.

PIX 535

The PIX 535 is the top-of-the-line model, suitable for service provider environments. Performance is the key: up to 1Gbps clear-text throughput, half a million simultaneous connections, and 7,000 connection initialization/teardowns a second. With the VAC, you can get 100Mbps 3DES throughput, with up to 2,000 simultaneous security associations (VPN tunnels).

In terms of hardware, the PIX 535 is based on a 1GHz Intel Pentium III, with up to 1GB of RAM. It has a 16MB flash and 256K cache running at 1GHz as well as a dual 64-bit 66MHz PCI system bus. Cards available are the one- or four-port 10/100 Ethernet NICs or 1GB Ethernet multimode "stick and click" fiber connectors.

The Console Port

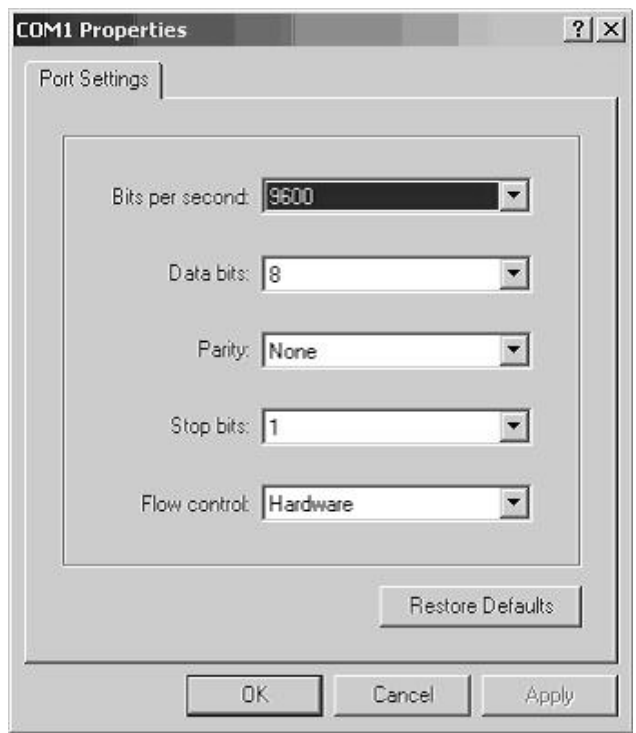
The primary mechanism for talking to a PIX is via the console port. Some devices have the old DB9 connectors—nine-pin D-subminiature connectors similar to those found on the back of many PCs. The newer devices use the Cisco standard RJ45 connector, similar to those found on their routers and switches. In each case, an appropriate cable is provided with your equipment.

The communication is via null-modem and uses communications set to 8-N-1. If you are using Windows, a good program to communicate with a PIX is Hyperterm, which is provided with most Windows-based installations, under Accessories/Communications. When launching Hyperterm, configure your connection to direct-connect to COM 1, as shown in Figure 2.5.

Figure 2.5 Configuring Hyperterm

The communications parameters then need to be set, as shown in Figure 2.6.

Figure 2.6 Port Communication Properties for Hyperterm



At this point, you should be connected. Power on your PIX, and you will see the boot process taking place, as shown in Figure 2.7. Your output will differ slightly.

Figure 2.7 Sample Output from Boot Sequence

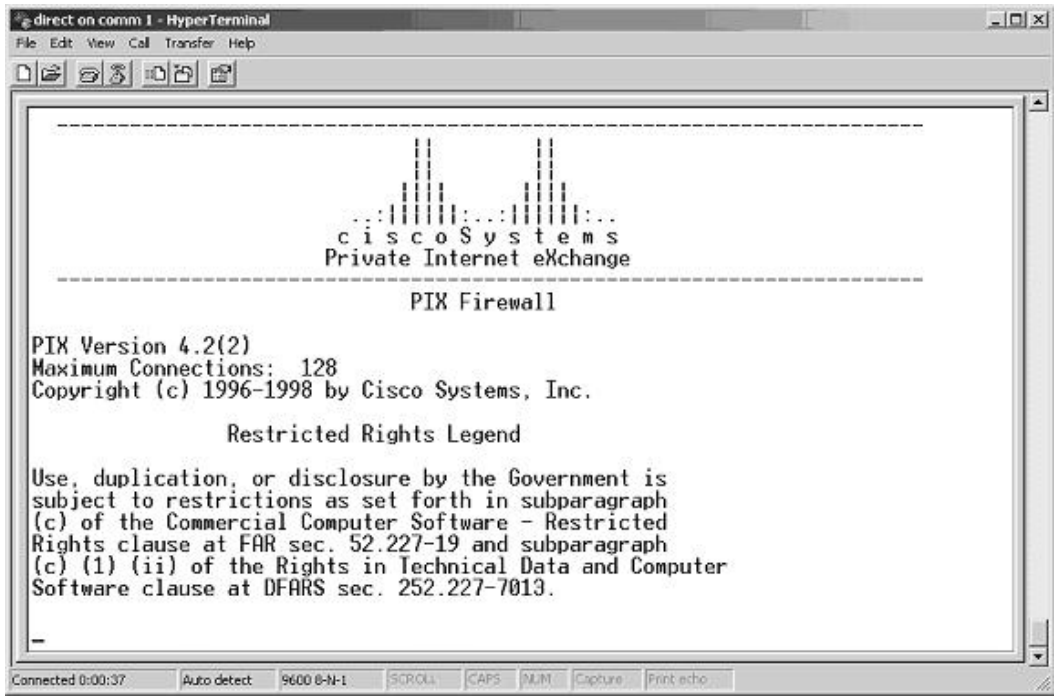


Figure 2.7 shows an older flash version, but they all are similar. If you do not see output or the output is garbled, it usually means your parameters are not set correctly. If you are not using the provided cable, make sure it is null-modem and that your parameters are set as shown in Figure 2.6.

Software Licensing and Upgrades

In order to have a flexible product, the PIX uses software licensing to enable or disable features within the PIX OS. Although the hardware is common to all platforms (except that certain licenses can ship with additional memory or hardware accelerators) and the software is common, features differ depending on the activation key.

The activation key allows you to upgrade features without acquiring new software, although the process is similar. The activation key is computed by Cisco depending on what you have ordered and your serial number, so it's different for

each piece of PIX hardware you own. The serial number is based on the flash, so if you replace the flash, you have to replace the activation key.

The activation key enables feature-specific information such as interfaces, high availability, and type of encryption. More specific information is found in the section on licensing.

To get information about the activation key, use the *show version* command. The command provides information about the code version, hardware information, and activation key information. Alternately, the command *show activation-key* provides something like this:

```
Serial Number: 480090153 (0x1c9d9829)


Running Activation Key: 0x75fe7c49 0xc08b4082 0x08979930 0xe4b4c4b0
Licensed Features:
Failover:             Enabled
VPN-DES:              Enabled
VPN-3DES:             Disabled
Maximum Interfaces:   6
Cut-through Proxy:    Enabled
Guards:               Enabled
URL-filtering:        Enabled
Inside Hosts:         Unlimited
Throughput:           Unlimited
IKE peers:            Unlimited
```

The *flash activation key* is the *same* as the running key.

This machine is a PIX 515 and has an unrestricted license, with the maximum number of interfaces permitted, including failover.

Updating the activation key in version 6.2 of the PIX OS couldn't be simpler. The command *activation-key <activation-key-four-tuple>* sets the key to the new value. Note that activation four-tuples are in hexadecimal, are case insensitive, and don't require you to start the numbers with 0x. Thus the previously mentioned machine could be set with:

```
PIX1(config)# activation-key 75fe7c49 c08b4082 08979930 e4b4c4b0
```

Updating the activation keys in prior versions is not much more complicated. Power-cycle the PIX, and send an Esc or Break to enter monitor mode. This will present you with a prompt:

```
monitor>
```

Type a **?** to see the options. Sample output is listed here:

```
Use ? for help.
monitor> ?
? this help message
address    [addr]     set IP address
file        [name]     set boot file name
gateway    [addr]     set IP gateway
help                  this help message
interface  [num]      select TFTP interface
ping        <addr>    send ICMP echo
reload                 halt and reload system
server      [addr]    set server IP address
tftp        TFTP      download
timeout     TFTP      timeout
trace                 toggle packet tracing
```

It would be a good idea to upgrade your software at this time, but in any event, the PIX will ask you if you want to update your activation key at the end of the TFTP process.

Licensing

Generally, the licensing falls into one of three types, plus an additional factor for crypto constraints. The three main categories are unrestricted, restricted, and failover. If you have a single PIX, you'll want unrestricted or restricted licensing, depending on the number of interfaces you want to support. If you have two PIX appliances and want high availability, you'll want one machine with an unrestricted license and another machine with a failover license.

Upgrading Software

The traditional way of managing images is via TFTP. This is a UDP-based transport protocol—fast and efficient. Unfortunately, it is not authenticated, so you have to be a bit careful to ensure that your data gets saved when you write to a TFTP server and that the data downloaded doesn't get corrupted.

By tradition, UNIX hosts have TFTP software preinstalled. If you do have a UNIX laptop, try *man tftpd* to see how to turn it on. If you have a Windows laptop, the server is not installed (although a client might well be—it's standard on most NT and Win2K environments).

Luckily, a TFTP server for a Windows environment is easy to acquire and install. Perhaps one of the best is the Solar Winds server, part of the Solar Winds suite. The full tool set is an invaluable aid to security professionals, and some pieces of it, like the TFTP server, are free. Installation is via the WISE installation wizard.

Another excellent TFTP server is the one Cisco provides. It is available at www.cisco.com/cgi-bin/tablebuild.pl/tftp and is also free. Simply provide your Cisco user ID when you download, and launch the installer executable.

Running the Cisco TFTP server is straightforward. The server, by default, is not running. (This mode is recommended, since there is no authentication; you don't want anyone uploading or downloading files without your knowledge.) The first time you run it, you will want to press **O** for Options (under the **View** menu) to set the log file, if desired, and set the TFTP root directory. This is where you want to store the images. If you are going to be upgrading the PIX software, FTP the binary image down from the Web into that directory, and you are ready for the transfer.

If you have a very old version of the software (pre 5.1(*x*)), you must upgrade using monitor mode. You can follow the preceding notes or the following step-by-step procedure:

1. Enter monitor mode. Remember, this requires that you get a console session running, power-cycle the box, and press **Escape** within 10 seconds of the boot.
2. The PIX is currently unconfigured. Set up your download interface by doing the following:
   - Use **interface <*number*>** to set the TFTP interface. The default is 1, so you don't have to set it if the TFTP server is on the inside.
   - Use **address <*IP address*>** to set the IP address of the PIX.
   - Hopefully, your server is on the same network as the TFTP interface. If not, you can set a default gateway with **gateway <*IP address*>**.
3. Next prepare the transfer information:
   - Use server <IP address> to set the IP address of your TFTP server.
   - Use file <filename> to set the name of the image to upload.
4. Finally, execute the transfer. Use tftp to start the file.

This process loads a new image in place, and when you reboot, you will come up under the new image.

Luckily, this process should not apply—unless you accidentally upload the wrong file or your TFTP transfer fails. Monitor mode is primarily used in the event of disaster.

The process of updating your software on a reasonably new version of code is straightforward. You can avoid monitor mode and do everything from the PIX enable command line. Log into the PIX and get into enable mode. It is a good idea to ping your TFTP server to verify connectivity—for example:

```
PIX1# ping inside 10.1.1.1
```

Get the version of the software onto your TFTP server, and copy the file to flash:

```
pixfirewall# copy tftp flash
Address or name of remote host [127.0.0.1]? 10.1.1.1
Source file name [cdisk]? pix621.bin
copying tftp://10.1.1.1/pix621.bin to flash
[yes|no|again]? yes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Received 1640448 bytes.
Erasing current image.
Writing 1640448 bytes of image.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Image installed.
```

On the next reload, the new image is available.

Password Recovery

Passwords are stored on the PIX using an MD5 hash. This is good; you are probably aware that Cisco type 7 passwords can be instantly decrypted using a simple personal digital assistant (PDA). MD5 hash is harder: A hacker essentially has to try out all the combinations. Unfortunately, the MD5 hash used on the PIX is significantly weaker than the Cisco type 5 hash used on Cisco routers. Programs such as Cain & Abel (www.oxid.it) can, with time, discover a password. This weakness has been assigned CVE vulnerability CAN-2002-0954. So if all you have is a printout, you can recover your password. This can be helpful for machines that are in production environments. (However, the caveat is that others can do the same. Be careful about leaving configuration files on TFTP servers or printouts where others can get to them.)

If your environment can tolerate a little downtime, you can reset your PIX password. You download a program, depending on your OS version, that will execute on the PIX and reset the password to the default, *cisco*. You can then get in and use enable mode to set the password to a known value.

Earlier you saw that monitor mode was used for emergencies. Forgetting the password is a pretty good emergency. Here is what you do:

1. Pick the correct version of the software from Table 2.2.

Table 2.2 PIX Password Recovery Binaries

Version	Filename	URL
4.3 and earlier releases	nppix.bin	www.cisco.com/warp/public/110/nppix.bin
4.4 release	np44.bin	www.cisco.com/warp/public/110/np44.bin
5.0 release	np50.bin	www.cisco.com/warp/public/110/np50.bin
5.1 release	np51.bin	www.cisco.com/warp/public/110/np51.bin
5.2 release	np52.bin	www.cisco.com/warp/public/110/np52.bin
5.3 release	np53.bin	www.cisco.com/warp/public/110/np53.bin
6.0 release	np60.bin	www.cisco.com/warp/public/110/np60.bin
6.1 release	np61.bin	www.cisco.com/warp/public/110/np61.bin
6.2 release	np62.bin	www.cisco.com/warp/public/110/np62.bin

2. Place this software on a TFTP server accessible to the PIX.
3. Connect to the PIX on the console port. Verify connectivity. (You should get a password prompt, which you can't answer.)
4. Reboot the PIX.
5. Within 10 seconds of the reboot, press **Esc** to enter monitor mode.
6. Use the *interface* command to set the interface to that of the TFTP server.
7. Use the *address* command to specify the IP address of that interface.
8. Use the *server* command to specify the IP address of the TFTP server.
9. Use the *gateway* command to specify the default route to the TFTP server, if needed. (This is not recommended; if at all possible, try to have the TFTP server on the same network as the PIX interface to minimize the likelihood of file corruption.)

10. Use the *file* command to specify the filename of the recovery file you chose in Step 1.
11. Use the *ping* command to verify that you can connect to the TFTP server.
12. Use the *tftp* command to start the download.

At this point, you should be prompted to erase the passwords, and you will be in. The default password has now been set to *cisco*, with no enable password.

The Command-Line Interface

Like a Cisco router, the configuration of the PIX is contained in a text file. The job of a PIX administrator is to create the text file. There are many ways to achieve this goal: working offline and uploading configurations, working through an intermediary such as the PIX Device Manager, or working at the command prompt. Because most maintenance tasks are fairly simple, most of your time will be spent at the command prompt, so it is helpful to spend some time with that.

Factory Default Configurations

There are two basic factory default configurations. Because the PIX 501 and PIX 506 have fairly specific purposes, the default configurations for those devices are suited to their market. Because the PIX 515, 525, and 535 are more general-purpose firewalls, they have correspondingly less configuration.

PIX 501 and 506E

The PIX 501 and 506E are intended to be dropped into a traditional DSL environment. Cisco makes the following assumptions:

1. The default information flow control policy will be anything permitted from the inside allowed out, nothing in.
2. The external interface will have its IP set via DHCP. Both interfaces are set fixed to 10Mbps Ethernet.
3. DHCP will be provided to inside users, with the default route set to the PIX.

The internal network that the PIX provides is the 192.168.1.0 network. (Remember, this is one of the choices allowed by RFC 1918.) The PIX will be the default gateway for the network, at 192.168.1.1. This is convenient since

many other vendors (such as wireless AP vendors) also use the 192.168.1.0 network and assume that the gate is at 192.168.1.1—so the 501 and 506E can be transparently dropped into most home nets. Limiting the interfaces to 10Mbps is not a problem, since the outside interface is going to be connected to a digital subscriber line (DSL) or cable environment, which will typically be functioning at less that 1Mbps, and fixing the connection to 10Mbps avoids some of the Fast Ethernet duplex handshaking problems that can occur on older switches.

For most users, this solution is reasonable. If this device is part of an enterprise deployment, a little more thought is required; this solution does not support centralized maintenance, for example, or VPN tunnels. If you are rolling out a large number of clients, you will want to determine a template and preconfigure the PIX before sending it to the end users.

PIX 515E, 525, and 535

The PIX 515E and up arrive with essentially blank factory configurations. Interfaces are set to autoconfigure but are disabled, and configuration via the console is required.

Administrative Access Modes

An administrative access mode is a state in which the administrator is able to issue commands, potentially to change the configuration of the PIX. Monitor mode, described earlier, is an administrative access mode, but it is contained in ROM rather than in the binary image, and hopefully you will never have to use it.

When you first log in, you are in an unprivileged mode. You can identify the mode you are in from the prompt: If the prompt looks like the hostname followed by a right-angle bracket (>), you are in unprivileged mode. Few commands are available:

```
PIX1> ?
enable          Turn on privileged commands
help            Help list
login           Log in as a particular user
logout          Exit from current user profile, and to unprivileged mode
pager           Control page length for pagination
quit            Quit from the current mode, end configuration or logout
```

This is not a complete list of the available commands. For example, when you are in unprivileged mode:

```
PIX1> show ?
checksum         View configuration information cryptochecksum
curpriv          Display current privilege level
history          Display the session command history
pager            Control page length for pagination
version          Display PIX system software version
PIX1> show version

Cisco PIX Firewall Version 6.2(1)
Cisco PIX Device Manager Version 1.0(1)

Compiled on Wed 17-Apr-02 21:18 by morlee

pix1 up 160 days 23 hours

Hardware:   PIX-515, 64 MB RAM, CPU Pentium 200 MHz
…
```

The most important of these is enable mode, which turns on the privileged commands. At this point, your prompt will change; now it ends in a pound sign. To show your new privilege:

```
PIX1# ?
arp         Change or view the arp table, and set the arp timeout value
capture     Capture inbound and outbound packets on one or more interfaces
configure   Configure from terminal
copy        Copy image or PDM file from TFTP server into flash.
debug       Debug packets or ICMP tracings through the PIX Firewall.
disable     Exit from privileged mode
eeprom      Show or reprogram the 525 onboard i82559 devices
flashfs     Show, destroy, or preserve filesystem information
help        Help list
kill        Terminate a telnet session
logout      Exit from current user profile, and to unprivileged mode
logging     Clear syslog entries from the internal buffer
pager       Control page length for pagination
passwd      Change Telnet console access password
ping        Test connectivity from specified interface to <ip>
quit        Quit from the current mode, end configuration or logout
```

```
reload      Halt and reload system
session     Access an internal AccessPro router console
shun        Manages the filtering of packets from undesired hosts
terminal    Set terminal line parameters
who         Show active administration sessions on PIX
write       Write config to net, flash, floppy, or terminal, or erase flash
```

At this point, you are more or less protected from accidentally harming the system: You can erase the configuration in total, but it will not make small changes until you enter configuration mode. Use the *configure terminal* command to get into configuration mode. Again, your prompt will change to show privilege:

```
PIX1(config)#
```

There are approximately 100 lines of commands, so it is not appropriate to show them all here. Unlike a Cisco router, for which there are additional modes, these are all the modes that occur: you have no rights, you are somewhat protected, or you are changing the configuration. However, note that if you are in configuration mode, your *show* commands are still available.

The PIX also stores previous commands you've executed. Use the *show history* command to see what you've executed. This feature is helpful in two ways: One, if you are unsure what you have executed so far, is to look at the *show history* command to see what you've done to date. A more common use is when you have lots of similar commands. You can use the Up Arrow key to see the previous line in your history and then use the basic commands (covered in the following section) to edit the line and resubmit it.

NOTE

The PIX firewall provides help functionality built into the command-line interface. Use the question mark key (?)—it is your friend. At any point, pressing **?** will help you complete your commands. In addition, a "man page" functionality is built in. For example, if you want to ping something and forgot the syntax, try *ping ?*. If you don't remember what the ping command does, try *help ping*. This provides not only usage but description and syntax issues.

Basic Commands

The environment at the command prompt is similar to that of a Cisco router and uses "emacs"-style commands, shown in Table 2.3.

Table 2.3 Basic Keystroke Shortcuts

Command	Result
Tab	Command-line completion.
Ctrl + A	Moves the cursor to the start of a line.
Ctrl + B	Moves the cursor one character left (nondestructive).
Alt + B	Moves the cursor one word left.
Ctrl + D	Deletes the character under the cursor.
Ctrl + E	Moves the cursor to the end of the line.
Ctrl + F	Moves the cursor one character right.
Alt + F	Moves the cursor one word right.
Ctrl + H or Rubout	Erases the previous character.
Ctrl + R	Reprints a line.
Up Arrow or Ctrl + P	Displays the previous line.
Up Arrow or Ctrl + N	Displays the next line.
Help or ?	Displays help.

To see additional editing commands, try searching the Web for *emacs style commands*. However, the list shown in Table 2.3 is very useful. For example, if you are setting up multiple ACL statements, you can save a great deal of effort by changing only a port number, then pressing **Ctrl + P** to get the previous line, **Alt + F** to move right a few words, **Ctrl + D** to delete the old port, then typing the new port.

In addition, you don't have to type the full command—you only have to provide enough of the command to establish a unique initial segment. For example, the command *configure terminal* can be abbreviated; the first three letters aren't enough (both *conduit* and *configure* start with *con*), and only one option from the *configure* command starts with *t*. So to get into configuration mode, just type *conf t*. Such shortcuts can save a bit of typing, particularly on long commands.

Hostname and Domain Name

Two useful commands are the *hostname* and *domain-name* commands. These set the hostname (which appears in the prompt) and the domain name of the PIX. The syntax is *hostname* <name> and *domain-name* <name>—for example:

```
PIX1 (config)# hostname PIX1
PIX1(config)# domain-name secret.com
```

Configuring Interfaces

The most important aspect of a network device is the network interface. In the PIX, configuring the network interface is a fairly straightforward process. You need to specify a few parameters to put the security in context and a few parameters to put connectivity in context, then the default information flow policy takes over.

The nameif Command

The *nameif* command is used to give an interface a logical name and assign it a security level. The name should be memorable, since it will be used in all other commands. The format of the *nameif* command is:

```
nameif <hardware_id> <interface> <security_level>
```

hardware_id corresponds to the hardware associated with the interface, such as ethernet0. *interface* corresponds to a descriptive name, such as *dmz*, and *security_level* corresponds to the level of trust, an integer between 100 (trusted) and 0 (untrusted).

The tradition is to put ethernet0 (the first card from the left) as the outside interface, with a security level of 0—for example:

```
PIX1(config)# nameif ethernet0 outside security0
```

To assign ethernet1 (the second card from the left) as the inside interface with a security level of 100, the command is:

```
PIX1(config)# nameif ethernet1 inside security100
```

The remaining cards, if any, are assigned values between 0 and 100. An example for a DMZ network might resemble the following:

```
PIX1(config)# nameif ethernet2 dmz security50
```

The interface Command

The *interface* command is used to set the physical layer properties of the interface. The syntax of the command is:

```
interface <hardware_id> <hardware_speed> [shutdown]
```

In this command, *hardware_id* corresponds to the value from the *nameif* command, and *hardware_speed* is chosen from Table 2.4.

Table 2.4 Hardware Speed Types for the interface Command

Value	Description
10baset	10Mbps Ethernet, half duplex.
100basetx	Fast Ethernet, half duplex.
100full	Fast Ethernet, full duplex.
1000sxfull	Gigabit Ethernet, full duplex.
1000basesx	Gigabit Ethernet, half duplex.
1000auto	Gigabit Ethernet to autonegotiate full or half duplex.
aui	10Mbps Ethernet, half duplex, for an AUI cable interface.
bnc	10Mbps Ethernet, half duplex, for a BNC cable interface.
auto	Sets Ethernet speed automatically. Generally, it is better to hardcode the cable type, since autonegotiation has failed with some hardware devices.

The optional *shutdown* keyword disables the interface; *shutdown* is useful to rapidly terminate a connection on a network that is at hazard or to ensure that unused networks are not accidentally added. An example of the *interface* command is:

```
PIX1(config)# interface ethernet0 100full
```

The ip address Command

The *ip address* command sets the IP address of the particular interface. The syntax of the command is as follows:

```
ip address <interface> <ip_address> <netmask>
```

In the *ip address* command, *interface* corresponds to the same parameter as in the *nameif* command, a descriptive term for the network, and *ip_address* and

netmask correspond to the usual properties for the interface. An example of this command might look something like this:

```
PIX1(config)# ip address dmz 192.168.0.1 255.255.255.0
```

NOTE

The PIX can also obtain an IP address through DHCP client or PPPoE functionality. These features are discussed in Chapter 4.

Static Routes

The PIX is not a router and so does not have a wide selection of routing protocols. The PIX supports static routes and RIP. Specifying a static route is done with the following syntax:

```
route <if_name> <ip_address> <netmask> <gateway_ip> [metric]
```

Translating this syntax into English, it reads "If packets destined for interface *if_name* on the network specified by network address *ip_address* are bounded by mask *netmask*, then route it via a next hop at *gateway_ip*." The optional *metric* command is used to give an indication of distance.

A particularly important route is the default route. This is the "route of last resort"—the route used when no other direction is known for the packet. Only one default route is allowed on the PIX. This route is indicated by the *0 route* with *netmask 0*; for example:

```
PIX1(config)# route outside 0 0 63.122.40.140 1
```

Password Configuration

Two passwords need to be set: a password for access to the PIX and an *enable* password to get into privileged (enable) mode. The PIX is limited to 16-byte passwords and is case sensitive. A basic *password* will assign a password, such as:

```
PIX1(config)# passwd cisco
PIX1(config)# enable password cisco
```

In the configuration, the password is stored in an encrypted fashion. The command then looks like this:

```
enable password 2KFQnbNIdI.2KYOU encrypted
passwd 2KFQnbNIdI.2KYOU encrypted
```

When first connecting to the PIX, you will see a password prompt:

```
Connected to 10.10.10.1.
Escape character is '^]'.


User Access Verification

Password:
Type help or '?' for a list of available commands.
pix1> en
Password: *****
```

You should note that to preserve security, the password is not echoed to the screen, and the previous sequence will get you into enable mode.

NOTE

The PIX also supports local user accounts with individual passwords. Alternatively, you can use RADIUS or TACACS+ for console authentication. You'll find a detailed discussion of these features in Chapter 5.

Managing Configurations

Just as with any network device, the most important task related to your PIX is ongoing management. It is important that you be comfortable not just manipulating the configuration with configuration mode but also pushing configurations out to storage and in from backup systems. Key commands here are *write*, which allows you to store a command; *copy*, which allows you to manage the underlying PIX application software, and *configure*, which allows you to update the configuration.

The write Command

The *write* command allows you to write the configuration to various types of media. Allowed variants are *write net*, *write memory*, *write standby*, *write terminal*, *write erase*, and *write floppy*.

```
write net [[server_ip] : [filename] ]
```

This command writes the configuration to a TFTP server. The IP address of the server can be specified on the command line or preset with the TFTP server command, *tftp-server [if_name] ip_address path*. Specifying a value on this line supercedes the value on the TFTP server line, but if the TFTP-server information is set, you can provide just a colon (or no parameters at all).

The next command allows you to store the configuration to flash. The *uncompressed* parameter specifies storing the configuration as an uncompressed string and is generally not necessary.

```
write memory [uncompressed]
```

If you want to print the configuration to the terminal (screen), use this command:

```
write terminal
```

Note that this command prints out the running configuration. In version 6.2, two new *show* commands were added: *show running-config*, which gives the same output as *write terminal*, and *show startup-config,* which shows the configuration that is written to flash. If the pager variable is set, the screen will pause after a fixed number of lines. To store the configuration via an ASCII capture, set the pager to 0, then type **write terminal**.

Similarly to the *write memory* command, on devices that have a diskette drive, the *write floppy* command stores the configuration in a proprietary format. This allows the PIX to readily read the configuration. If you write the configuration to a PIX boot disk, the appliance will come up with the desired configuration. Unfortunately, it is not easily readable on other devices.

```
write floppy [uncompressed]
```

There is one other *write* command: *write erase*. This command clears the flash configuration to a known good state and allows you to reconfigure.

The copy Command

The *copy* command is a similar way of managing images. The most common use of the command is in the *copy tftp* command—for example:

```
copy tftp[:[[//location] [/tftp_pathname]]] flash[:[image | pdm]]
```

The first couple of parameters are straightforward: They deal with specifying the location and filename of the TFTP server and, as previously mentioned, can

be set with the *TFTP-server* command. The keyword *flash* indicates that the information is being stored to flash. The files can be conventional images, in which case they are available on the next reload, or PDM images, in which case they are available immediately.

Images can also be downloaded from a Web server via conventional HTTP or over SSL. This is specified by the following command:

```
copy http[s]://[user:password@] location [:port ] / http_pathname flash
    [: [image | pdm] ]
```

You can probably figure out the parameters. The first part is the standard URI notation: *http* for clear-text Web use or *https* for SSL service. The *user:password@location* portion allows you to encode user information; if you are working via a Web browser, this portion triggers a popup window asking you to fill in your username and password. Since the PIX does not have a popup, you can specify it on the command line by inserting it before the @ sign. If the Web server is running on a nonstandard port, you can also specify it here by putting the port after a colon, similar to this:

```
copy http://fwadmin:cisco@10.10.10.1:99/pix_image flash
```

This solution is convenient if you do not have a TFTP server handy and can safely store the image files on a Web server.

The configure Command

You can manage configurations via the *configure* command. This is often the dual to the *write* commands. For example, just as *write terminal* dumps the configuration to the terminal, *configure terminal* allows you to change the configuration from the terminal.

These commands generally merge the configuration from the media with the existing configuration. You will often want to *clear configure* to wipe out the existing configuration so you can pull a complete stored config. The other choices are:

```
configure [terminal|floppy|memory]
```

You've used this one already, in the *conf t* command. It allows you to add commands from the terminal, from a diskette (if the PIX has a diskette drive), or from flash (memory).

Analogous to the *copy* command, this command:

```
configure http[s]://[<user>:<password>@]<location>[:<port>]/<pathname>
```

merges a configuration that is stored on a Web server with the running configuration.

```
configure net [<location>]:[<pathname>]
configure factory-default [<inside_ip> [<mask>]]
```

Resetting the System

Generally, after fetching a new image, you will want to have the PIX start under the new image. Similarly, it is helpful to occasionally restore the configuration to what is running on the flash—if, for example, you have been exploring commands and have gotten to an uncertain state. You can always power-cycle the device; this solution has no moving parts, and configurations and images are fully flushed to flash, so you do not have to worry about corruption. But there is a better way: the *reload* command.

The reload Command

You can restart the PIX gracefully using the *reload* command. This command prompts you, to ensure that you really mean what you are saying; it can only be executed from privileged mode:

```
pix1# reload
Proceed with reload? [confirm]
```

At this point, there is a brief pause while the PIX reboots, and then you will be working under the new system. Note: If you want to bypass pressing the second carriage return, you can type **reload noconfirm**, but when you are executing a potentially dangerous command such as a reboot, it is generally good to have an "Are you really sure you want to do this?" checkpoint.

Summary

The PIX is a dedicated firewall appliance based on a special-purpose, hardened operating system. The simplified kernel and reduced command structure (compared with firewalls based on general-purpose operating systems) means that all other things being equal, the PIX will have higher throughput and more reduced maintenance costs than the general-purpose device. In addition, the similarity to IOS provides an edge to security administrators who are familiar with the Cisco environment.

The PIX is a hybrid firewall based on stateful packet filtering with the use of proxies for specific applications. The stateful packet filter is known as the *Adaptive Security Algorithm,* or *ASA,* and uses two databases: a table of translations and a table of known connections, to maintain state of the traffic transiting the network and to dynamically allow packets through the filter. The ASA inspects both packet header information, including source address, destination address, and TCP and UDP socket information, as well as packet contents for certain protocols, to make intelligent decisions on routing the packets. ASA has additional features: It will rewrite packets where necessary, as part of its inspection engine, where the protocols are well known.

About a dozen proxies are associated with the PIX. Some, such as the FTP proxy, augment the ASA process by permitting the passing of packets associated with an allowed communication—for FTP, while the command channel follows the normal three-way handshake initiated by the client and directed at a well-known socket, the data channels have the handshake initiated by the server (in the opposite direction of the usual security policy) and directed at a port defined during the transaction. Others, such as the SMTP proxy, are designed to enforce a limited subset of protocol commands and, by enforcing the RFC, provide additional security to potentially buggy applications. Still others, such as the multimedia proxies, provide the intelligence to extract IP addresses from the body of the packets and handle the complex rewriting and authorization for these interrelated protocols.

In addition to its native packet-filtering and access control features, the PIX provides additional common firewall services. Again, a key advantage of an appliance is performance, and the PIX makes an excellent VPN terminator, with the ability to pass encrypted traffic at wire speed, when an accelerator card is installed. It can provide content logging and filtering to help control Web surfing and provides address translation to allow for either "sewing together" networks seamlessly at the perimeter or consolidating (and concealing) internal networks to present to the outside world a limited number of addresses.

Modern environments depend on firewalls, and so the PIX provides high resiliency through its failover mechanism. This mechanism provides for a *hot spare*—a second PIX with an equivalent configuration that will automatically press itself into service should the primary device fail.

The PIX's extensive capabilities are matched by hardware flexibility. As of this writing, five different models are shipping, designed to match almost any environment. The PIX 501 is designed for the SOHO user, with a small switch built in for basic use. The PIX 506E, designed for the small or branch office, supports better performance for connecting back to the corporate hub. The PIX 515E is designed for the enterprise core of small to medium-sized business, with a rack-mount chassis and corresponding enterprise-class performance. The PIX 525 is designed for large enterprise or small service provider environments and has a slot-based configuration to allow for multiple interface configurations. The PIX 535 is the top-of-the-line model, designed for service provider environments, with the best possible throughput of the PIX appliances.

Communicating with an unconfigured PIX is most easily achieved through the console cable. This is provided with each firewall kit. Use a communications program such as Hyperterm, set your parameters to 8-N-1, and during the boot sequence you will see characters on your screen.

Licensing for the PIX features is set via an activation key. You should have received information about your activation key when you purchased the PIX; additional features can be purchased and new activation keys applied. The activation keys are dependent on a (hardware) serial number based on your flash. You can add new keys through either monitor mode or the *activation-key* command, new to version 6.2. Licensing usually falls into three types: unrestricted (all features enabled), restricted (limited features and interfaces), or failover (used for hot standby machines).

Password recovery is achieved by running a special program (different for each version of the operating system) on the PIX itself. The process requires either a dedicated boot diskette or the use of monitor mode and a TFTP download of a temporary image.

The normal configuration of the PIX is achieved through a command-line interface. This interface uses the "emacs" editing commands and is very similar to that provided in the Cisco IOS. The command structure is modal, with three major modes: unprivileged, which has very few available commands; privileged, where all commands are available (subject to your privilege level, which can be set in a local database), and configuration mode, by which changes are made to the running configuration.

Things that you will want to set up in every configuration include host and domain name, which configures the prompt and controls fields in the digital certificates used in VPN traffic, and the properties of the interfaces. You control a name—an association between a distinctive identifier for the interface and its default security characteristics—physical properties, and IP properties. You will also probably want to set up some basic routing, particularly the default route.

Passwords on any security device are very important. There are passwords for access to the device (unprivileged mode) and for escalation to privileged mode. They can be shared passwords, one per box, or passwords on a per-user basis. Cisco recommends the latter method, which requires setting up AAA services, either remote or local.

Managing configuration information is also important. Once you have built the perfect config, you do not want to have to retype it all in case of an emergency. Configurations can be stored in human-readable format via an ASCII capture (via *write terminal*) or as a text file on a TFTP server (via *write net*). Images can also be brought onto the system with the *copy* command, either from a TFTP server (*copy tftp*) or from a Web server URL (*copy https://*servername/pix_image *flash*). The system can then be restarted with the *reload* command and is ready to run under the new configuration.

Solutions Fast Track

PIX Firewall Features

- ☑ The PIX is based on a dedicated operating system with security functionality as the focus as opposed to being just another feature of the general operating system.
- ☑ All other things equal, a dedicated operating system will provide higher throughput since fewer other tasks are being performed and will have lower maintenance costs because there are fewer patches to manage.
- ☑ The heart of the PIX's functionality is the stateful packet filter, known as the Adaptive Security Algorithm, or ASA. It is a dedicated procedure that manages state, contained in two key databases, the CONN table and XLATE table.

- ☑ The PIX is a hybrid firewall, combining packet filtering with dedicated proxies for specialized protocols such as H323 and SMTP. It contains other protective features such as fragment protection and DNS replay protection.
- ☑ In addition to "classic" firewall features, the PIX has several other features: VPN termination (helpful for integrating encrypted traffic into the firewall policies); URL filtering (helpful because the firewall sits in a choke point and is a natural place to do filtering), and NAT/PAT capabilities so that the firewall can conceal internal address structures and extend the available address space.

PIX Hardware

- ☑ The PIX line has five models, designed for deployments ranging from home users to service provider firewall cores.
- ☑ The PIX 501 has a desktop form factor and is designed for the SOHO environment. It is designed to transparently drop into home user networks without requiring user configuration, but it supports central administration and all the features of the rest of the product line.
- ☑ The PIX 506E is similar to the 501 but is aimed at the small or branch office deployments. It also has an easy setup and is designed to support more users.
- ☑ The PIX 515E is designed for the enterprise core and is a rack-mounted appliance. It is also suitable as an internal firewall, isolating internal enterprise departments.
- ☑ The PIX 525 is designed for large enterprise use. It is rack-mountable, like the 515E, but has the capability for multiple interface configurations such as providing service networks in addition to the Internet and trusted networks.
- ☑ The PIX 535 is the highest-performing appliance, aimed at the service provider environment. It combines the highest performance of the PIX product line with the flexibility of the 525 line.

PIX Software Licensing and Upgrades

- ☑ The PIX license is based on an activation key. The key is unique to your serial number, which is tied to the hardware flash. If you replace the flash, you need a new key.
- ☑ Licensing is soft-upgradeable via installation of a new activation key. This allows for a "pay as you need it" approach, allowing for new features such as 3DES encryption or additional interfaces to be enabled only as required, without having to replace the hardware.
- ☑ Licensing is sold as one of three types: unrestricted, restricted, failover. Failover provides high availability (with a second hardware device). Restricted licensing limits the number of connections or interfaces, whereas unrestricted licensing allows all features to be enabled.

The Command-Line Interface

- ☑ The command-line interface uses "emacs style" commands. This allows for manipulating the command line to cycle through the command history as well as editing the existing line.
- ☑ Under normal operation, the command line is in one of three modes: unprivileged, privileged, and configuration. Unprivileged mode allows you to inspect a limited number of parameters and to access privileged mode. Privileged mode allows you full access to the commands without changing the configuration. As its name suggests, configuration mode is the way you actually update the device environment.
- ☑ The PIX supports a variety of configuration management technologies: The config can be written to flash or out to TFTP servers. Since the configurations are textual in nature, they can be read or manipulated outside the PIX similarly to any text file.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: Can I manage the PIX remotely without using Telnet?

A: Yes. Starting with version 5.3, check out SSH compatibility. The service is enabled with:

```
ssh ip_address [netmask] [interface_name]
```

You need a DES or 3DES activation key and must manage an RSA key pair. Since Telnet passes passwords in the clear, its use is deprecated except on very tightly controlled networks, and the use of SSH or console access is encouraged.

Q: Does the PIX support SNMP management?

A: Yes. The PIX supports read-only SNMP access via the *snmp-server* commands. You can set a community string and trigger traps to a collection agent.

Q: Does the PIX support syslog-style events?

A: Yes. The PIX supports event management via syslog; multiple syslog hosts can be specified. See the *logging* command for more details. (Earlier versions used the *syslog* commands; they are retained for backward compatibility, but you should migrate to using the term *logging*.)

Q: Will the PIX provide DHCP services?

A: Yes. In fact, DHCP is enabled by default on the 501 and 506 devices. DHCP is a service that will dynamically assign IP addresses to (internal) hosts as they boot up. This service allows laptop users to automatically acquire IP addresses. In smaller environments, enabling DHCP allows for additional convenience in managing networks.

For higher-end PIX devices, DHCP is available but not turned on by default. Although this is not a security hole, it is helpful to control DHCP via a separate device, so that the IP-to-MAC address mappings can be monitored, queried, and otherwise controlled.

Q: I am trying to allow <*protocol*> from the Inside to DMZ1. I opened the appropriate port, and for the first person, everything worked great. The instant a second inside person tries to use <*protocol*>, however, everything breaks. What can I do?

A: Check your fixups. If <*protocol*> matches a fixup, you might be able to adjust performance. If that is not working, it might be a NAT/PAT problem. Remember that PAT remaps IP addresses and port numbers to a single unique IP address. Some protocols will not permit that kind of remapping; switching from PAT to NAT could help. Better yet, see if you can avoid the use of NAT altogether. When you are mapping from inside to the DMZ, you are probably using private addresses on both sides. Turn off the NAT translation, and you should be able to pass traffic safely.

Chapter 3

Passing Traffic

Solutions in this chapter:

- **Allowing Outbound Traffic**
- **Allowing Inbound Traffic**
- **TurboACLs**
- **Object Grouping**
- **Case Study**

- ☑ **Summary**
- ☑ **Solutions Fast Track**
- ☑ **Frequently Asked Questions**

Introduction

A firewall would not serve any purpose if it blocked all traffic. To properly protect a network environment, network traffic must be filtered both outbound and inbound. The key to configuring a firewall is to ensure that it only allows the traffic you want allowed and only blocks the traffic you want blocked. In some cases, this is not an easy task.

In this chapter, you will learn how to pass traffic through the PIX firewall. To pass traffic through a PIX firewall, some form of address translation must be configured. You will learn how to set up both static and dynamic translations. Once translation has been configured, the PIX will automatically allow all responses by default. To configure more granular access, you can permit or deny specific traffic, using access lists and conduits. Depending on whether you are configuring inbound or outbound access, different commands are available to accomplish this task. We discuss these different commands in this chapter.

Object grouping is a new feature in PIX firewalls that simplifies access list configuration and maintenance. We will discuss how to create and use object groups.

Throughout the chapter, we use examples to describe the various commands. We provide a complex case study to review what you have learned. By the end of this chapter, you will be an expert on passing traffic through PIX firewalls.

Allowing Outbound Traffic

Once the initial configuration is complete, the first step to pass traffic is allowing outbound access. This requires configuring address translation or explicitly disabling it. Once address translation is configured, and unless an access list or apply/outbound list prohibits it, all outbound traffic is allowed by default. This is a primary feature of the Adaptive Security Algorithm (ASA) and is the reason that security levels are so critical. The PIX maintins state information on each connection. This enables responses that arrive on a lower security interface to be sent to the recipient on a high security interface.

Configuring Dynamic Address Translation

Address translation is necessary to pass outbound traffic. Address translation (through NAT and/or PAT) maps local IP addresses to global IP addresses.

Configuration of NAT/PAT is a two-step process:

1. Identify the local addresses that will be translated (*nat* command).
2. Define the global addresses to translate to (*global* command).

Address translation records are called *translation slots* (or *xlate*) and are stored in a table known as the *translation table*. To view the contents of this table, use the *show xlate* command. The *xlate* timer monitors the translation table and removes records that have been idle longer than the defined timeout. By default, this timeout is set to three hours by default.

The syntax of the *nat* command is as follows:

```
nat [(<if_name>)] <id> <local_address> [<netmask> [outside] [dns]
     [norandomseq] [timeout <hh:mm:ss>] [<connection_limit>
          [<embryonic_limit>]]
```

The *if_name* parameter is used to apply the *nat* command to the interface where the traffic to be translated enters the PIX. This parameter must match the name assigned to the interface with the *nameif* command. If no name is specified, the inside interface is assumed.

The *id* parameter is an integer between 0 and 2,000,000,000 that that links the local IP addresses (*local_address*) identified by the *nat* command to the global IP addresses specified by the *global* command. The *id* 0 is special as it specifies addresses that are not to be translated. The local address will be the global address.

The *netmask* parameter is used with *local_address* to specify subnets or multiple IP addresses. The *outside* keyword specifies external addresses to be translated. The *dns* keyword translates IP addresses in DNS responses using active entries in the translation table. By default, when performing address translation, the PIX firewall randomizes the sequence numbers. The *norandomseq* keyword tells the PIX not to randomize the sequence numbers. This is useful when you will be performing address translation twice (for example, when you have two PIX firewalls in the path) and do not need randomization twice. The *timeout* parameter defines how long to allow an entry in the translation table to stay idle.

The *connection_limit* parameter defines how many concurrent active connections are allowed, and the *embryonic_limit* parameter defines how many concurrent half-open connections are allowed. Half-open connections indicate a TCP connection that hasn't completed the handshaking process. Both of these parameters default to 0, allowing unlimited connections. Excessive half-open connections can be the result of a DoS attack. Tuning *embryonic_limit* can reduce the impact of these attacks.

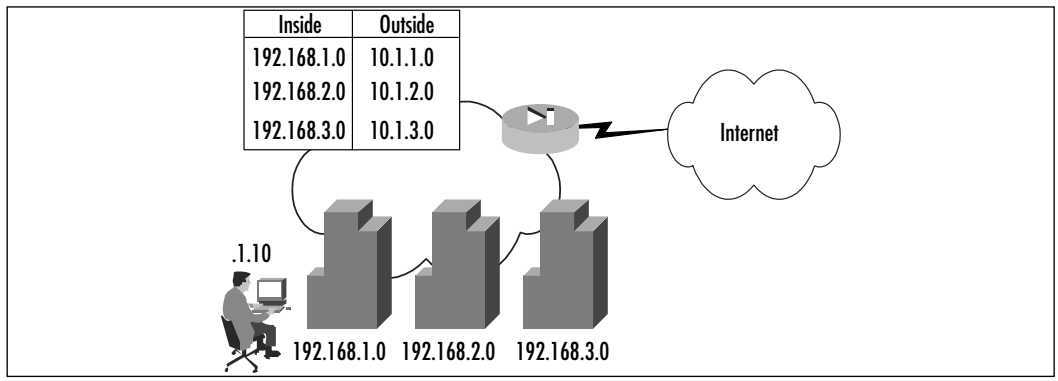
The *global* command defines the pool of addresses to be used for translation. These are typically public addresses. The syntax for the *global* command is as :

```
global [(<if_name>)] <id> { {<global_ip> [-<global_ip>] [netmask
    <global_mask>]} | interface}
```

The *if_name* parameter defines the interface on which traffic will exit after being translated. If it is not specified, the outside interface is assumed. The *id* parameter links *global* to one or more *nat* statements. The *global_ip* parameter defines the IP addresses to translate local addresses. If a single IP address is specified, port address translation (PAT) is performed. If a range is specified, network address translation (NAT) is used until no more global addresses are available. Once all global addresses have been exhausted, PAT is performed. The *netmask* keyword is compiled with *global_ip* to derive the range of IP addresses. The *interface* keyword allows local addresses to be translated to an existing interface address, and to an alternative to *global_ip*.

Let's look at the ficticious Secure Corporation, a company that has decided to network three buildings in London and provide Internet access to its employees. This company does not own any IP addresses of its own. One of the company's requirements is to use private address space, because it does not want to readdress the entire network if it has to change ISPs. By utilizing a private IP address scheme, the company can change public IP addresses whenever circumstances require. All it will have to do is associate the new IP address range to the private IP addresses. Figure 3.1 shows the network layout. (*Note:* Even though it is a private address range, the 10.0.0.0/8 network is being used to represent the public IP address space in this chapter. Keep this in mind as you read the rest of the chapter.)

Figure 3.1 A Network Address Translation Example



In Figure 3.1, you can see that each of the three buildings has been assigned a 24-bit network from the private address range specified in RFC 1918. These ranges are 192.168.1.0/24, 192.168.2.0/24, and 192.168.3.0/24, respectively. Each ISP-assigned 24-bit subnet (10.1.1.0/24, 10.1.2.0/24, and 10.1.3.0/24) has been mapped to a private address range. This configuration allows each node to have a unique public IP address dynamically mapped from a pool associated with the originating building. The configuration in this example is fairly straightforward. Traffic to be translated must be identified using the *nat* command and then mapped to a pool of public IP addresses defined by the *global* command. The commands to configure this are as follows:

```
PIX1(config)# nat (inside) 1 192.168.1.0 255.255.255.0
PIX1(config)# global 1 10.1.1.1-10.1.1.254 netmask 255.255.255.0
PIX1(config)# nat (inside) 2 192.168.2.0 255.255.255.0
PIX1(config)# global 2 10.1.2.1-10.1.2.254 netmask 255.255.255.0
PIX1(config)# nat (inside) 3 192.168.3.0 255.255.255.0
PIX1(config)# global 3 10.1.3.1-10.1.3.254 netmask 255.255.255.0
PIX1(config)# exit
PIX1# clear xlate
```

NOTE

The *clear xlate* command clears contents in the translation table. This command should be executed after any translation configuration changes are made; otherwise, there is a danger of stale entries remaining in the translation table.

To make sure that everything was entered correctly, use the *show nat* and *show global* commands:

```
PIX1# show nat
nat (inside) 1 192.168.1.0 255.255.255.0 0 0
nat (inside) 2 192.168.2.0 255.255.255.0 0 0
nat (inside) 3 192.168.3.0 255.255.255.0 0 0
PIX1# show global
global (outside) 1 10.1.1.1-10.1.1.254 netmask 255.255.255.0
global (outside) 2 10.1.2.1-10.1.2.254 netmask 255.255.255.0
global (outside) 3 10.1.3.1-10.1.3.254 netmask 255.255.255.0
```

The ISP provided enough public addresses that Secure Corp. was able to create a one-to-one mapping between local and global addresses. What would happen if the ISP did not allocate enough public address space? Let's assume that the ISP provided a single 24-bit public address range (10.1.1.0/24). Instead of using multiple address pools, the company could use one global pool for all buildings and use PAT. PAT, as explained in Chapter 1, enables many-to-one address translation. The following configuration initially performs NAT, then PAT once there are no available addresses:

```
PIX1(config)# nat (inside) 1 192.168.1.0 255.255.255.0
PIX1(config)# nat (inside) 1 192.168.2.0 255.255.255.0
PIX1(config)# nat (inside) 1 192.168.3.0 255.255.255.0
PIX1(config)# global (outside) 1 10.1.1.1-10.1.1.254 netmask 255.255.255.0
PIX1(config)# exit
PIX1# clear xlate
```

NOTE

PAT works with DNS, FTP, HTTP, mail, RPC, rsh, Telnet, URL filtering, and outbound *traceroute*. PAT does not work with H.323, caching name servers, and PPTP.

To enable NAT on multiple interfaces, use separate *global* commands on each interface. Use the same *id* on all the *global* commands. This allows a single set of *nat* commands on the target interface to translate private (local) IP addresses to

one of many different global address ranges based on destination. The following commands configure the PIX to NAT the 192.168.1.0/24 network to either a 10.1.1.0/24 address or PAT to the DMZ interface IP address, depending on the interface the packet will exit:

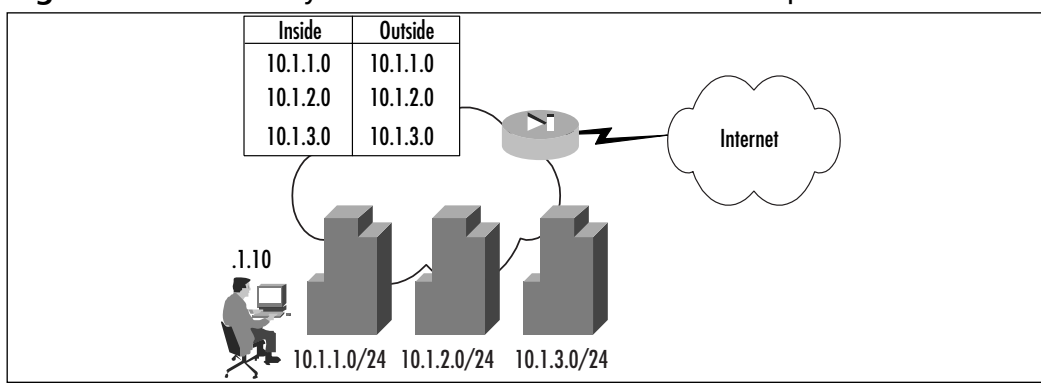
```
PIX1(config)# nat (inside) 1 192.168.1.0 255.255.255.0
PIX1(config)# global (outside) 1 10.1.1.1-10.1.1.254 netmask 255.255.255.0
PIX1(config)# global (dmz) 1 interface
PIX1(config)# exit
PIX1# clear xlate
```

As with most commands on the PIX firewall, use the *no* keyword with the *nat* and *global* commands to remove them from the configuration.

Identity NAT and NAT Bypass

Suppose our Secure Corp. decided not to use private IP addresses inside the PIX, and chose to use public IP addresses. Secure Corp. has been assigned a block of public IP addresses from the American Registry for Internet Numbers (ARIN) in the form of three 24-bit networks. The corporation chooses, as shown in Figure 3.2, not to use private addressing within its network.

Figure 3.2 An Identity Network Address Translation Example



Looking at Figure 3.2, you can see that each of the three 24-bit subnets has been allocated to each building. Public addresses will be used both inside and outside the PIX firewall, and no address translation will be performed. There are two ways to accomplish this task: using identity NAT or using NAT bypass.

Identity NAT does not use an associated *global* command to define the global address. Instead, the internal address is mapped to itself when translating. To configure identity NAT, use the *nat* command with an *id* of 0. Do not define an associated *global* command. The commands to configure identity NAT in Figure 3.2 would be as follows:

```
PIX1(config)# nat (inside) 0 10.1.1.0 255.255.255.0
nat 0 10.1.1.0 will be non-translated
PIX1(config)# nat (inside) 0 10.1.2.0 255.255.255.0
nat 0 10.1.2.0 will be non-translated
PIX1(config)# nat (inside) 0 10.1.3.0 255.255.255.0
nat 0 10.1.3.0 will be non-translated
PIX1(config)# exit
PIX1# clear xlate
```

Configuring & Implementing…

Identifying "All" Network Traffic

Instead of using specific networks to identify the traffic to translate using the *nat* command, you can use a source address of 0 or 0.0.0.0 and a netmask of 0 or 0.0.0.0 to specify *all* traffic.

To verify the configuration, use the *show nat* command to view the current NAT configuration:

```
PIX1# show nat
nat (inside) 0 10.1.1.0 255.255.255.0 0 0
nat (inside) 0 10.1.2.0 255.255.255.0 0 0
nat (inside) 0 10.1.3.0 255.255.255.0 0 0
```

Let's examine the example in Figure 3.2. The client opens a connection to a Web server on the Internet. The *show xlate* command should show a mapping for this connection flagged with an *I*, or identity flag.

```
PIX1# show xlate debug
1 in use, 1 most used
Flags: D - DNS, d - dump, I - identity, i - inside, n - no random,
     o - outside, r - portmap, s - static
NAT from inside:10.1.1.10 to outside:10.1.1.10 flags iI idle 0:01:27
     timeout 3:00:00
```

You can also bypass NAT altogether using *nat 0* with an access list. First, define an access list that identifies the traffic to be translated (access lists are discussed in detail in the next section). Then, use the *nat* command with an *id* of 0 and the access list name to bypass the NAT process. The syntax to configure this is:

```
access-list <acl_name> permit ip <source_addr> <source_mask> <dest_addr>
     <dest_mask>
nat (<if_name>) 0 access-list <acl_name>
```

Using Figure 3.1 as an example, the commands to configure the PIX to bypass NAT using an access list would be as follows:

```
PIX1(config)# access-list inside_public permit ip 10.1.1.0 255.255.255.0
     any
PIX1(config)# access-list inside_public permit ip 10.1.2.0 255.255.255.0
     any
PIX1(config)# access-list inside_public permit ip 10.1.3.0 255.255.255.0
     any
PIX1(config)# nat (inside) 0 access-list inside_public
PIX1(config)# exit
PIX1# clear xlate
```

To verify the configuration, use the *show nat* and *show access-list* commands:

```
PIX1# show nat
nat (inside) 0 access-list inside_public
PIX1# show access-list
access-list inside_public; 3 elements
access-list inside_public permit ip 10.1.1.0 255.255.255.0 any (hitcnt=0)
access-list inside_public permit ip 10.1.2.0 255.255.255.0 any (hitcnt=0)
access-list inside_public permit ip 10.1.3.0 255.255.255.0 any (hitcnt=0)
```

In Figure 3.2, when the client opens a connection to a Web server on the Internet, the *show xlate* command should not show a translation for this connection since it bypasses NAT. The *show access-list* command should show an incremented *hitcnt* counter on the appropriate access list entry.

```
PIX1# show xlate
0 in use, 1 most used
PIX1# show access-list inside_public
access-list inside_public; 3 elements
access-list inside_public permit ip 10.1.1.0 255.255.255.0 any (hitcnt=10)
access-list inside_public permit ip 10.1.2.0 255.255.255.0 any (hitcnt=0)
access-list inside_public permit ip 10.1.3.0 255.255.255.0 any (hitcnt=0)
```

Although identity NAT and NAT bypass provide similar functionality, using NAT bypass provides some advantages over identity NAT. These advantages include saving resources by bypassing the NAT process and greater flexibility specifying destination addresses in the access list.

Blocking Outbound Traffic

If certain outbound traffic needs to be blocked, this must be done explicitly. Controlling when outbound traffic is allowed to traverse the PIX firewall is always a part of a well-designed security policy. There are two ways to accomplish this task: using access lists or using outbound/apply statements. Access lists, introduced in PIX firewall software version 5.0, are the newer and recommended method for controlling outbound access. Only use outbound/apply statements if you have to (for example, if you have an older version of the PIX software).

Access Lists

Access lists on the PIX firewall are very similar to those used on Cisco routers and can be used to limit the traffic allowed to transit the PIX based on several criteria, including source address, destination address, source TCP/UDP ports, and destination TCP/UDP ports. Access list configuration is a two-step process:

1. Create the ACL permit and deny statements using the *access-list* command.
2. Apply the access list to an interface using the *access-group* command.

There are two different syntaxes for the *access-list* command. The first is used for any protocol other than Internet Control Message Protocol (ICMP), and the second is used for ICMP:

```
access-list <acl_name> {deny | permit} <protocol> <src_addr> <src_mask>
    [<dest_operator> <dest_port>] <dest_addr> <dest_mask> [<dest_operator>
        <dest_port>]
access-list <acl_name> {deny | permit} icmp <src_addr> <src_mask>
    <dest_addr> <dest_mask> <icmp_type>
```

The *acl_name* parameter identifies the access list and can be either a name or a number. The *permit* and *deny* keywords are self-explanatory. The *protocol* parameter specifies the IP protocol. You can either enter the numerical value or specify a literal name. Possible literal names are listed in Table 3.1.

Table 3.1 Literal Protocol Names and Values

Literal	Value	Description
ah	51	Authentication header for IPv6, RFC 1826
eigrp	88	Enhanced Interior Gateway Routing Protocol
esp	50	Encapsulated Security Payload for IPv6, RFC 1827
gre	47	General Routing Encapsulation
icmp	1	Internet Control Message Protocol, RFC 792
igmp	2	Internet Group Management Protocol, RFC 1112
igrp	9	Interior Gateway Routing Protocol
ip	0	Internet Protocol
ipinip	4	IP-in-IP encapsulation
nos	94	Network Operating System (Novell's NetWare)
ospf	89	Open Shortest Path First routing protocol, RFC 1247
pcp	108	Payload Compression Protocol
snp	109	Sitara Networks Protocol
tcp	6	Transmission Control Protocol, RFC 793
udp	17	User Datagram Protocol, RFC 768

The address of the network or host from which the packet originated is specified using the *src_addr* parameter. The *src_mask* parameter specifies the netmask bits to apply to *src_addr*. To specify all networks or hosts, use the *any* keyword,

which is equivalent to a source network and mask of 0.0.0.0 0.0.0.0. Use the *host* keyword followed by an IP address to specify a single host. The *dest_addr* and *dest_mask* are similar to the *src_addr* and *src_mask* parameters, except that they apply to destination addresses.

NOTE

The syntax for access lists on the PIX firewall is very similar to that of Cisco routers. The key difference is that access lists on PIX firewalls use standard wildcard masks, whereas on routers they use inverse wildcard masks. For example, when blocking a 24-bit subnet, you would use a mask of 255.255.255.0 on a PIX firewall and a mask of 0.0.0.255 on a Cisco router.

An operator comparison lets you specify a port or port range and is used with the *tcp* or *udp* protocol keywords. To specify all ports, do not specify an operator and port. Use *eq* to specify a single port. Use *gt* to specify all ports greater than the specified port. Use *neq* to specify all ports except a given number. Finally, use *range* to define a specific range of ports. The port can be specified using either a number or a literal name. A list of literal port names is presented in Table 3.2.

Table 3.2 Literal Port Names and Values

Name	Port	Protocol
bgp	179	tcp
biff	512	udp
bootpc	68	udp
bootps	67	udp
chargen	19	tcp
citrix-ica	1494	tcp
cmd	514	tcp
daytime	13	tcp
discard	9	tcp/udp
dnsix	195	udp

Continued

Table 3.2 Continued

Name	Port	Protocol
domain	53	tcp/udp
echo	7	tcp/udp
exec	512	tcp
finger	79	tcp
ftp	21	tcp
ftp-data	20	tcp
gopher	70	tcp
h323	1720	tcp
http	80	tcp
hostname	101	tcp
ident	113	tcp
irc	194	tcp
isakmp	500	udp
klogin	543	tcp
kshell	544	tcp
login	513	tcp
lpd	515	tcp
mobile-ip	434	udp
nameserver	42	udp
netbios-dgm	138	udp
netbios-ns	137	udp
nntp	119	tcp
ntp	123	udp
pim-auto-rp	496	tcp/udp
pop2	109	tcp
pop3	110	tcp
radius	1645, 1646	udp
rip	520	udp
smtp	25	tcp
snmp	161	udp
snmptrap	162	udp
sqlnet	1521	tcp

Continued

Table 3.2 Continued

Name	Port	Protocol
sunrpc	111	tcp/udp
syslog	514	udp
tacacs	49	tcp/udp
talk	517	tcp/udp
telnet	23	tcp
tftp	69	udp
time	37	udp
uucp	540	tcp
who	513	udp
whois	43	tcp
www	80	tcp
xdmcp	177	tcp

Note that the system-defined port mapping of *http* is the same as *www* and is silently translated in the configuration. The *icmp_type* parameter allows you to permit or deny access to ICMP message types. A list of ICMP message types can be found in Table 3.3.

Table 3.3 ICMP Message Types

ICMP Type	Literal
0	echo-reply
3	unreachable
4	source-quench
5	redirect
6	alternate-address
8	echo
9	router-advertisement
10	router-solicitation
11	time-exceeded
12	parameter-problem
13	timestamp-reply
14	timestamp-request

Continued

Table 3.3 Continued

ICMP Type	Literal
15	information-request
16	information-reply
17	mask-request
18	mask-reply
31	conversion-error
32	mobile-redirect

After configuring the access list, you must apply it to an interface using the following command:

```
access-group <acl_name> in interface <if_name>
```

The name associated with an access list is specified as *acl_name,* whereas the name of the interface that the access list will use to monitor inbound traffic is specified by *if_name*. An applied access list denies or permits traffic as it enters the PIX on the specified interface.

NOTE

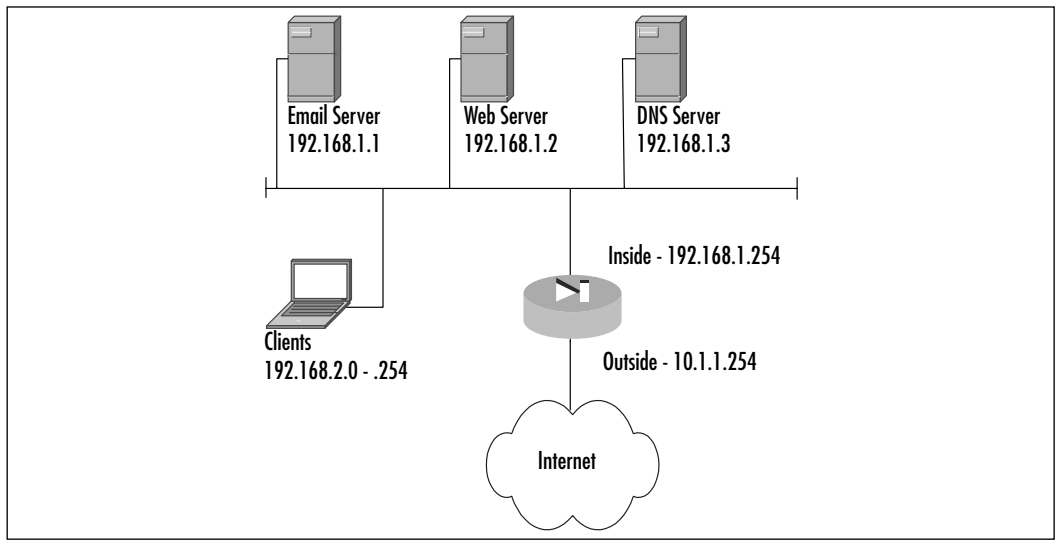
Access lists on the PIX firewall can only be applied to traffic *entering* an interface, not traffic that is exiting an interface. This is unlike Cisco routers, on which access lists can be applied in either direction.

Access lists on the PIX firewall have an implicit *deny all* at the end. This means that unless traffic has been specifically permitted within the access list, it will be denied by the implied deny-all that follows the last entry in every access list. This provides additional security by assuming that traffic not explicitly recognized is to be denied. If there are errors in the configuration, the wrong traffic may be permitted or denied. Since access lists are processed sequentially from top to bottom, a PIX administrator can create very complex access lists simply by following the flow of what should and should not be allowed. Only one access list at a time can be applied to an interface.

Let's now look at Secure Corp., which has just purchased a new PIX firewall for its network in New York, as shown in Figure 3.3. All the servers that the company hosts at the site, as well as all the clients within the network, are located

on the inside interface of the PIX. The site uses a single network with the address space of 192.168.0.0/22. The ISP has assigned the 10.1.1.0/24 public network to use.

Figure 3.3 The Secure Corporation Access List Example



The company's requirements are that the clients only be able to access the Internet with their Web browsers. Company servers may have unrestricted access to the Internet. The design of an access list should start with a definition of what is going to be allowed and then proceed to what is going to be denied. In this example, the access list will have to allow clients in the 192.168.2.0/24 range to access any Internet server on TCP port 80. Then, the access list will allow the three listed servers unfettered access to the Internet. The following commands accomplish this result:

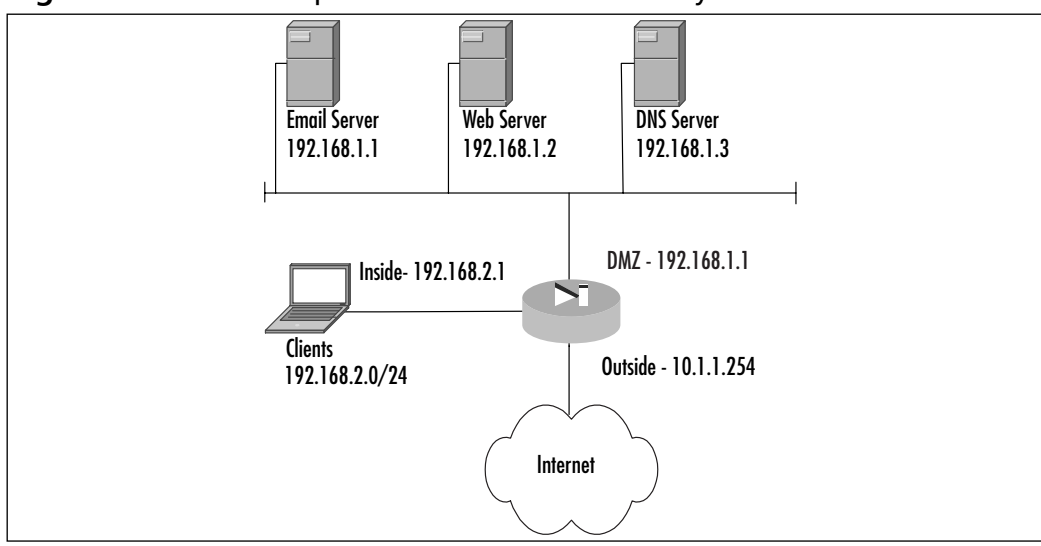
```
PIX1(config)# access-list inside_in permit tcp 192.168.2.0 255.255.255.0 any
    eq 80
PIX1(config)# access-list inside_in permit ip 192.168.1.1 255.255.255.255 any
PIX1(config)# access-list inside_in permit ip 192.168.1.2 255.255.255.255 any
PIX1(config)# access-list inside_in permit ip 192.168.1.3 255.255.255.255 any
PIX1(config)# access-group inside_in in interface inside
```

A good practice is to add an explicit *deny all* statement to the end of an access list so you remember it is there when you do a *show access-list* command. You can see how many packets have been dropped using the *hitcnt* counter:

```
PIX1(config)# access-list inside_in deny ip any any
PIX1(config)# exit
PIX1# show access-list
access-list inside_in; 4 elements
access-list inside_in permit tcp 192.168.2.0 255.255.255.0 any eq www
    (hitcnt=2)
access-list inside_in permit ip host 192.168.1.1 any (hitcnt=0)
access-list inside_in permit ip host 192.168.1.2 any (hitcnt=0)
access-list inside_in permit ip host 192.168.1.3 any (hitcnt=0)
access-list inside_in deny ip any any (hitcnt=40)
```

Best security practices dictate that publicly accessible servers should not be located on the inside network; instead, they should be located on a DMZ network. The DMZ provides an extra layer of security and controls the risks associated with a publicly accessible server. If the server becomes compromised, it is possible to contain the compromise to the DMZ and still protect inside clients. However, if the network is set up as in the previous example and the server becomes compromised, there is very little that can be done to stop that server from compromising the entire internal network (you can shut the server down or disconnect it). Keep this design practice in mind. Figure 3.4 shows a revised network layout.

Figure 3.4 Secure Corporation Revised Network Layout



It is apparent that the network requirements have changed, because services the clients used to access without going through the firewall now need to be added to the access lists. Unlike the access list created previously, the servers should not be allowed to access any IP address without restriction. A DMZ access list should be created that limits the services that the servers are able to use. If these servers become compromised, you want to limit their infection of your networks. The commands to create and apply these access lists are:

```
PIX1(config)# access-list inside_in permit tcp 192.168.2.0 255.255.255.0
    any eq www
PIX1(config)# access-list inside_in permit tcp 192.168.2.0 255.255.255.0
    192.168.1.1 eq smtp
PIX1(config)# access-list inside_in permit tcp 192.168.2.0 255.255.255.0
    192.168.1.1 eq pop3
PIX1(config)# access-list inside_in permit udp 192.168.2.0 255.255.255.0
    192.168.1.3 eq domain
PIX1(config)# access-list inside_in permit tcp 192.168.2.0 255.255.255.0
    192.168.1.3 eq domain
PIX1(config)# access-list inside_in deny ip any any
PIX1(config)# access-group inside_in in interface inside
PIX1(config)# access-list dmz_in permit tcp 192.168.1.1 255.255.255.255
    any eq smtp
PIX1(config)# access-list dmz_in permit udp 192.168.1.3 255.255.255.255
    any eq domain
PIX1(config)# access-list dmz_in permit tcp 192.168.1.3 255.255.255.255
    any eq domain
PIX1(config)# access-list dmz_in deny ip any any
PIX1(config)# access-group dmz_in in interface dmz
```

It is important to note that we have not yet covered how to configure inbound access. The preceding access list only allows these servers to initiate contact with other servers—as a client would do. For example, the e-mail server can send mail to another domain, but it cannot receive it. The DNS server can resolve domain information from another domain, but it cannot respond to queries from other domains. The "Allowing Inbound Traffic" section of this chapter covers in detail how inbound access is enabled.

One very useful feature in configuring the PIX is the *name* command. This command allows you to define a name alias to an IP address so that during

configuration, instead of referencing a host by its IP address, the host can be referenced by a name. This feature is useful during complex configurations, because a descriptive name eases configuration and troubleshooting. The syntax for the command is:

```
name <ip_address> <name>
```

For example, the following command maps the name *mail* to the IP address 10.1.1.10:

```
PIX1(config)# name 10.1.1.10 mail
```

The name *mail* can now be used in access lists instead of an IP address. When you delete a name entry, all references to it in an access list revert to the IP addresses. Be sure the name statement is the last thing you remove during a clean-up.

Outbound/Apply

The *outbound* and *apply* commands control what traffic is allowed to exit the network. The *outbound* command only identifies traffic to be permitted or denied. The *apply* command puts the outbound list on an interface and actually causes packets to be dropped. The first step to control outbound traffic is configuring *outbound* to identify the traffic to be filtered. The syntax for the *outbound* command is:

```
outbound <list_id> permit | deny <ip_address> [<netmask> [<port>[-<port>]]
    [<protocol>]
```

The *list_id* is an identifier that maps the traffic identified by the *outbound* command to the *apply* command; *list_id* must be a number between 1 and 99. The *permit* or *deny* keywords specify whether the traffic identified by the *outbound* command will be permitted or denied, respectively. The *ip_address* parameter specifies the traffic to be identified by the *outbound* command. The *netmask* parameter is used in conjunction with the *ip_address* parameter to identify target IP address ranges. The *port* parameter specifies a specific port number or range to be identified by the *outbound* command. The *protocol* parameter identifies specific protocols (*tcp, udp*, etc.) and is assumed to be *ip* if it is not specified.

The second step is to apply the *outbound* list to an interface using the *apply* command. Once applied to an interface, any outgoing traffic to that interface is denied by the associated *outbound* list will be dropped. The syntax for the *apply* command is as follows:

```
apply [(<if_name>)] <list_id> outgoing_src | outgoing_dest
```

The *interface_name* parameter identifies the interface on which traffic will be filtered with the associated *outbound* list. If no interface is specified, it defaults to the outside interface. The *list_id* parameter names the *outbound* list to use for filtering outbound traffic. Unlike access lists, multiple outbound lists can be applied to an interface. These lists are processed starting at the lowest number and working upwards. This list is read top to bottm and is cumulative.

The *outgoing_src* or *outgoing_dest* keywords define how the *apply* command uses the *outbound* list. If *outgoing_src* is used, the *ip_address* is a source address. If *outgoing_dest* is used, it is a destination address.

Returning to Secure Corp., the company has decided to restrict access from its networks to the Internet. To control what employees can access, the company has decided to deny all packets from the company to *echo, chargen*, and *discard* services on the Internet. They chose these ports because they are common ports for attacking Internet servers. There is no reason an employee should need access to these services on an outside host.

To accomplish this task, create an outbound list. Configure this list to allow all traffic through. Next, define rules that deny access to the specific services. Finally, apply the *outbound* list to an interface. The commands to accomplish these tasks are as follows:

```
PIX1(config)# outbound 20 permit 0.0.0.0 0.0.0.0 0
PIX1(config)# outbound 20 deny 0.0.0.0 0.0.0.0 echo
PIX1(config)# outbound 20 deny 0.0.0.0 0.0.0.0 discard
PIX1(config)# outbound 20 deny 0.0.0.0 0.0.0.0 chargen
PIX1(config)# apply (inside) 20 outgoing_src
```

Unfortunately, even after taking all these precautions, the company receives a complaint that an employee is attempting to access a server on the Internet that they should not. The IP address of the Internet server that is being illegally accessed is 10.10.1.10. A new outbound rule needs to be created. Since the company can't figure out which employee is causing the problem, instead of filtering traffic by the source address, use the *apply* command to filter by the destination:

```
PIX1(config)# outbound 30 permit 0.0.0.0 0.0.0.0 0
PIX1(config)# outbound 30 deny 10.10.1.10 255.255.255.255 0
PIX1(config)# apply (inside) 30 outgoing_dest
```

Another way to accomplish this is to use the *outbound* command with the *except* keyword. The *except* keyword reverses the outbound list direction for the specified IP address. For example, if the rule specified source addresses, *except* would make a specific destination be denied. In the preceding example, instead of creating a new outbound list, we could add an *except* parameter to outbound list 20:

```
PIX1(config)# outbound 20 except 10.10.1.10 255.255.255.255 0
```

To verify your configuration, use the *show outbound [list_id]* command.

NOTE

It might be desirable to block Java applets or ActiveX code arriving from the Internet. The PIX supports this functionality. For more information, refer to Chapter 4, which provides detailed information on URL, Java, and ActiveX filtering.

Allowing Inbound Traffic

Up to this point in the chapter, we have not discussed how to allow traffic from an untrusted host to a server protected by the PIX. The PIX would not be entirely functional to most organizations if it did not allow traffic from an untrusted source to contact servers such as a corporate Web server. The PIX ASA treats traffic transiting a lower security-level interface to a higher security-level interface (*inbound traffic*) differently than outbound traffic.

Unlike outbound traffic, inbound traffic is denied by default. This is to ensure that the security levels of the interfaces are respected and not bypassed. As with outbound traffic, allowing inbound traffic to traverse the PIX is a two-step process. First, configure (static) translation. Second, configure an access list or conduit to specifically allow the inbound traffic. Similar to the *outbound/apply* commands, the *conduit* command has been superceded by access lists.

Static Address Translation

With a publicly accessible server (ideally located in a DMZ), you must explicitly allow connections from the lower security-level interface to a higher security-level interface. First, create a static address translation. The *static* command creates a permanent mapping of global-to-local IP addresses. The syntax for the command is as follows:

```
static [(<internal_if_name>, <external_if_name>)] {<global_ip> |
    interface} <local_ip> [netmask <mask>] [<max_conns> [<em_limit>]]
        [norandomseq]
```

The *static* command requires two arguments: the internal interface (interface to which the server being translated is connected), and the external interface, (where the global IP address is assigned). The *global_ip* and *local_ip* parameters are self-explanatory. The *netmask* parameter is used to statically translate more than one IP address at a time**.** The default value for both *max_conns* and *em_limit* is 0 (unlimited); these have meaning as they do in the *nat* command.

Secure Corp. has added a DMZ network to its PIX. It has decided to move its Internet Web server to this DMZ and permit access to it from the Internet. Figure 3.4 shows the network layout. The *static* commandto configure this follows:

```
PIX1(config)# static (dmz, outside) 10.1.5.10 192.168.1.2 netmask 255.255
    .255.255 0 0
```

If Secure Corp. had more than one Web server, instead of configuring a separate static entry for each one, you could configure a single *static* command with the correct netmask. For example, for 14 Web servers that had the IP addresses of 192.168.1.1 through 192.168.1.15, you would use the following command:

```
PIX1(config)# static (dmz, outside) 10.1.5.0 192.168.1.0 netmask 255.255
    .255.240 0 0
```

The Web server in the DMZ needs to access a database server located on the inside network of the PIX. The database server IP address does not need to be translated, since the Web servers on the DMZ are a part of the private address network. The following *static* configuration translates the IP address to itself. This is similar to *nat 0*:

```
PIX1(config)# static (inside, dmz) 192.168.1.2 192.168.1.2 netmask 255.255
    .255.255 0 0
```

We are now halfway to allowing inbound traffic access to a protected server. The *static* command only creates a static address mapping between global and local IP addresses. Since the default action for inbound traffic is to deny it, the next step is to create an access list or conduit to allow the traffic to enter the PIX. Like the *outbound/apply* commands, the *conduit* command became a legacy command in favor of access lists when version 5.0 of the PIX software was released.

Access Lists

The process of creating an access list to allow inbound access is similar to the process of creating an access list for outbound access, which was discussed earlier in this chapter. The command syntax is the same, as are all the parameters. Static translation must be configured to enable the lower security level traffic to access the higher security-level networks.

Conduits

Using *conduits* is another method for allowing inbound access. Its syntax is provided here:

```
conduit permit | deny <protocol> <global_ip> <global_mask> [<operator>
    <port> [<port>]] <foreign_ip> <foreign_mask> [<operator> <port>
        [<port>]]
```

Cisco recommends not using conduits, but to use access lists instead. The *protocol, operator*, and *port* parameters are the same as in access lists. The *global_ip* parameter defines the global IP addresses of the host to allow or deny access to, and the *foreign_ip* parameter defines the IP address to allow access from. The *global_mask* and *foreign_mask* parameters are the subnet masks applied to *global_ip* and *foreign_ip,* respectively.

The PIX processes the *conduit* commands in the order they are typed. Once conduits have been created, nothing more has to be done to enable them. Conduits are not explicitly applied to an interface. Based on the *global_ip*, conduits are applied to source and destination addresses.

For example, if a Web server with an internal IP address of 172.16.1.10 resides on the DMZ network, the following commands would allow access to it from any foreign IP address:

```
PIX1(config)# static (dmz, outside) 10.1.5.10 172.16.1.10 netmask 255.255
    .255.255 0 0
PIX1(config)# conduit permit tcp host 10.1.5.10 eq www any
```

Since the Web server is using a private IP address, the foreign client would use the public address to access the server. The conduit created would only work between the outside and DMZ interfaces because the *static* command defines these interfaces in the translation.

Another example of conduit commands is as follows. This command enables DNS lookups to occur from anywhere outside the network to the DNS server with address 10.1.5.11:

```
PIX1(config)# static (dmz, outside) 10.1.5.11 172.16.1.11 netmask 255.255
    .255.255 0 0
PIX1(config)# conduit permit udp host 10.1.5.11 eq domain any
PIX1(config)# conduit permit tcp host 10.1.5.11 eq domain any
```

This command enables an e-mail server (172.16.1.12) to receive SMTP e-mail from outside the network as 10.1.5.12:

```
PIX1(config)# static (dmz, outside) 10.1.5.12 172.16.1.12 netmask 255.255
    .255.255 0 0
PIX1(config)# conduit permit tcp host 10.1.5.12 eq smtp any
```

The *show conduit* command, as illustrated here, can show all the conduits currently configured on the PIX:

```
PIX1# show conduit
conduit permit tcp host 10.1.5.10 eq www any (hitcnt=0)
conduit permit udp host 10.1.5.11 eq domain any (hitcnt=0)
conduit permit tcp host 10.1.5.11 eq domain any (hitcnt=0)
conduit permit tcp host 10.1.5.12 eq smtp any (hitcnt=0)
```

ICMP

Inbound ICMP traffic can be controlled using the *icmp* command, which only filters ICMP traffic terminating on one of the PIX interfaces, not traversing the PIX. The command has the following syntax:

```
icmp {permit|deny} <ip_address> <netmask> [<icmp_type>] <if_name>
```

The ip_*address* parameter is the source address of the ICMP packet that will be denied or permitted. The *netmask* parameter is the mask associated with the *ip_address* parameter. The *icmp_type* parameter specifies the ICMP type to be denied or permitted. A list of the ICMP type values was presented earlier in Table 3.3. The *if_name* parameter is the interface to which this ICMP filter will be applied.

The following command permits the DMZ interface to respond to pings from network 172.16.0.0 255.255.240.0:

```
PIX1(config)# icmp permit 172.16.0.0 255.240.0.0 echo dmz
```

Port Redirection

Port redirection allows one public IP address to serve as the public IP address for more than one server. Port redirection allows you to define a mapping between a port on a public IP address and a port on a private IP address. To enable redirection, an access list or conduit must be created, as traffic is crossing from a lower security-level interface to a higher security-level interface.

Mappings can be set at the port level, and an IP address can serve many servers. Secure Corp. has set up a network at its Toronto site and assigned only a single public IP address from the ISP. At this site, Secure Corp. has two Web servers, one Telnet server, and one FTP server. How can it make all these services accessible publicly with a single IP address? Use the *static* command to perform port redirection:

```
static [(<prenat_if_name>, <postnat_if_name>)] {tcp | udp} {<global_ip>
    | interface} <global_port> <local_ip> <local_port> [netmask <mask>]
        [<max_conns> [<em_limit>]] [norandomseq]
```

We discussed the *static* command earlier in the chapter, so we will not go through all the parameters again. However, we will introduce some new parameters here, including *global_port* and *local_port*. A protocol (*tcp* or *udp*) must also be specified so that the PIX knows the protocol/port pair to accept and forward. Instead of using a *global_ip*, you can use the *interface* option to specify the IP address of the PIX interface in *postnat_if_name*. This option is important if you do not have any additional public IP addresses.

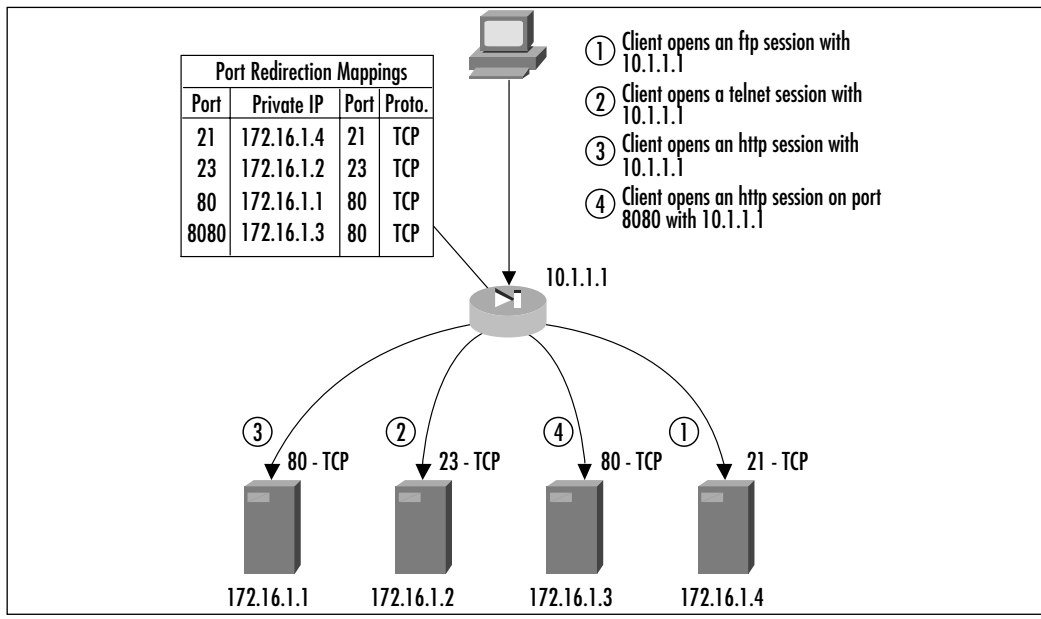
To configure port redirection for the first Web server, the command is as follows:

```
PIX1(config)# static (dmz, outside) tcp interface 80 172.16.1.1 80
```

If the company also wanted to host Telnet, FTP, and another Web server, three more *static* commands would map the ports to the correct servers. Since the Web port is already taken, a high port (8080) is chosen for access to the second Web server. This example is shown in Figure 3.5. The additional commands are as follows:

```
PIX1(config)# static (dmz, outside) tcp interface 23 172.16.1.2 23
PIX1(config)# static (dmz, outside) tcp interface 8080 172.16.1.3 80
PIX1(config)# static (dmz, outside) tcp interface 21 172.16.1.4 21
```

Figure 3.5 A Port Redirection Example



TurboACLs

TurboACLs are a new feature in PIX firewall software version 6.2. The general principal behind TurboACLs is that a long or complex access list is *compiled,* or indexed, to enable faster processing of the access list.

TurboACLs do not speed up short access lists. The PIX will not enable this feature on an access list unless it is over 18 lines. With longer access lists, the TurboACL feature creates an index (something like that in a book) that enables the PIX to process the long access list more quickly.

The index created by a TurboACL consumes a fair amount of resources. For this reason, Cisco recommends that TurboACLs should not be configured on anything lower than a 525 series firewall. To enable the TurboACL feature on all access lists of the PIX, use the *access-list compiled* command, as shown:

```
PIX1(config)# access-list compiled
```

To verify that the TurboACLs are turned on, issue a *show access-list* command:

```
PIX1(config)# show access-list
access-list compiled
access-list inside_public turbo-configured; 3 elements
access-list inside_public permit ip 10.1.1.0 255.255.255.0 any (hitcnt=0)
access-list inside_public permit ip 10.1.2.0 255.255.255.0 any (hitcnt=0)
access-list inside_public permit ip 10.1.3.0 255.255.255.0 any (hitcnt=0)
```

If you choose not to enable them at a global level, TurboACLs can be turned on and off for individual access lists. This feature can be useful if you only have a few access lists that need to be optimized. To configure a single access list to use the TurboACL feature, the syntax is:

```
access-list <acl-name> compiled
```

If a PIX has more than one access list, and only access lists applied to the outside interface need the TurboACL feature, turn it off except on the outside interface shown:

```
PIX1(config)# no access-list compiled
PIX1(config)# access-list outside_in compiled
```

Object Grouping

Introduced in PIX software version 6.2, *object grouping* makes very complex access lists much simpler to configure. Before object-grouping, each unique network, node, service, and protocol combination defined in an access list had to be configured with a separate *access-list* statement. However, in most organizational security policies, groups of entries have similar access rights. Object groups allow groups of network addresses, services, protocols, and ICMP types to be defined, reducing the number of access list entries.

For example, if an organization wants to deny access to several external FTP servers, they had to deliver an access list entry for each individual FTP server.

Using object groups, we can define a network object group containing the IP addresses of the banned FTP servers. IP addresses can easily be added and removed from this group. Only one access list entry has to be created denying access to the object group. The access list does not need to be modified if entries are added or removed from the object group. Object groups simplify access list configuration and maintenance.

Configuring and Using Object Groups

There are four types of object groups: *icmp-type, protocol, network,* and *service*. Each object group type corresponds to a field in the *access-list* or *conduit* command. Once an object group has been created, a subconfiguration mode is entered to populate the group. Each object group type has different subconfiguration options, so we will look at each separately. Once an object group has been configured, it can be used in an *access-list* or *conduit* command.

ICMP-Type Object Groups

An *ICMP-type object group* is a group of ICMP types (numerical or literal). ICMP-type object groups can be used in place of the *icmp-type* parameter in an access list or conduit. To create an ICMP-type object group:

```
object-group icmp-type <grp_id>
```

Once an object group has been defined, the subconfiguration mode enables the object group to be populated. An optional description can be specified using the *description* subcommand. The syntax is as follows:

```
icmp-object <icmp_type>
```

The following object group defines ICMP-type values to be used later in an access list or conduit:

```
PIX1(config)# object-group icmp-type icmp-grp
PIX1(config-icmp-type)# description ICMP Type allowed into the PIX
PIX1(config-icmp-type)# icmp-object echo-reply
PIX1(config-icmp-type)# icmp-object unreachable
PIX1(config-icmp-type)# exit
PIX1(config)# exit
```

Network Object Groups

A *network object group* is a group of IP addresses or networks. Network object groups can be used in place of a *src_addr* or *dst_addr* parameter in an access list or conduit statement. To create a network object group, the syntax is as follows:

```
object-group network <grp_id>
```

Network object groups have two subcommands for defining the group of hosts and networks. The syntax for defining a host entry in the object group is:

```
network-object host <host_addr | host_name>
```

The *host_addr* parameter is the IP address of the host being added to the object-group. The *host_name* parameter specifies the hostname of a host defined through the *name* command.

The syntax for defining a network entry in the object group is:

```
network-object <net_addr> <netmask>
```

The following object group defines hosts and networks to be used later in an access list or conduit:

```
PIX1(config)# object-group network net-grp
PIX1(config-network)# description List of Public HTTP Servers
PIX1(config-network)# network-object host 192.168.1.10
PIX1(config-network)# network-object host 172.16.10.1
PIX1(config-network)# network-object 172.16.2.0 255.255.255.0
PIX1(config-network)# exit
PIX1(config)# exit
```

Protocol Object Groups

A *protocol object group* is a group of protocol numbers or literal values. Protocol object groups can be used instead of the *protocol* parameter in an access list or conduit. To create a protocol object group, the syntax is as follows:

```
object-group protocol <grp_id>
```

Once an object group has been defined, the subconfiguration mode enables the object group to be populated as shown:

```
protocol-object <protocol>
```

The following object group defines a group of protocols that will be used later in an access list or conduit to provide VPN access:

```
PIX1(config)# object-group protocol vpn-grp
PIX1(config-protocol)# description Protocols allowed for VPN Access
PIX1(config-protocol)# protocol-object ah
PIX1(config-protocol)# protocol-object esp
PIX1(config-protocol)# protocol-object gre
PIX1(config-protocol)# exit
PIX1(config)# exit
```

Service Object Groups

A *service object group* is a group of TCP or UDP port numbers. Service object groups can be used in place of the *port* parameter in an access list or a conduit. The syntax to create a service object group is as follows:

```
object-group service <grp_id> tcp|udp|tcp-udp
```

Since a service object group lists ports and port ranges, they need to be configured as TCP, UDP, or both. The *tcp, udp*, and *tcp-udp* keywords define the common IP protocol for all ports listed in the object group. The subconfiguration command to populate the service object group with a single port is:

```
port-object eq <port>
```

The subconfiguration command syntax to populate the service object group with a range of ports is:

```
port-object range <begin-port> <end-port>
```

The following object group defines a group of ports that all Web servers within in organization need to have opened on the firewall:

```
PIX1(config)# object-group service websrv-grp tcp
PIX1(config-service)# description Ports needed on public web servers
PIX1(config-service)# port-object eq 80
PIX1(config-service)# port-object eq 8080
PIX1(config-service)# port-object range 9000 9010
```

To verify that an object group was created and populated with the correct information, we can view the current object group configuration using the *show object-group* command:

```
PIX1# show object-group
object-group icmp-type icmp-grp
  description: ICMP Type allowed into the PIX
  icmp-object echo-reply
  icmp-object unreachable
object-group network net-grp
  description: List of Public HTTP Servers
  network-object host 192.168.1.10
  network-object host 172.16.10.1
  network-object 172.16.2.0 255.255.255.0
object-group protocol vpn-grp
  description: Protocols allowed for VPN Access
  protocol-object ah
  protocol-object gre
  protocol-object esp
object-group service websrv-grp tcp
  description: Ports needed on public web servers
  port-object eq www
  port-object eq 8080
  port-object range 9000 9010
```

If one of the object groups does not look correct or is not needed, it can be removed using the *no object-group <grp_id>* command.

While object groups can be used in access lists and conduits, they must be preceded by the *object-group* keyword. To allow the ICMP type values defined in the *icmp-grp* object group, the *access-list* command is:

```
PIX1(config)# access-list icmp_in permit icmp any any object-group icmp-grp
```

To allow access to the Web servers defined in the *net-grp* on the ports defined in *websrv-grp,* the command is:

```
PIX1(config)# access-list outside_in permit tcp any object-group net-grp
    object-group websrv-grp
```

One nice feature of object groups is that they can nest object groups of the same type. For example:

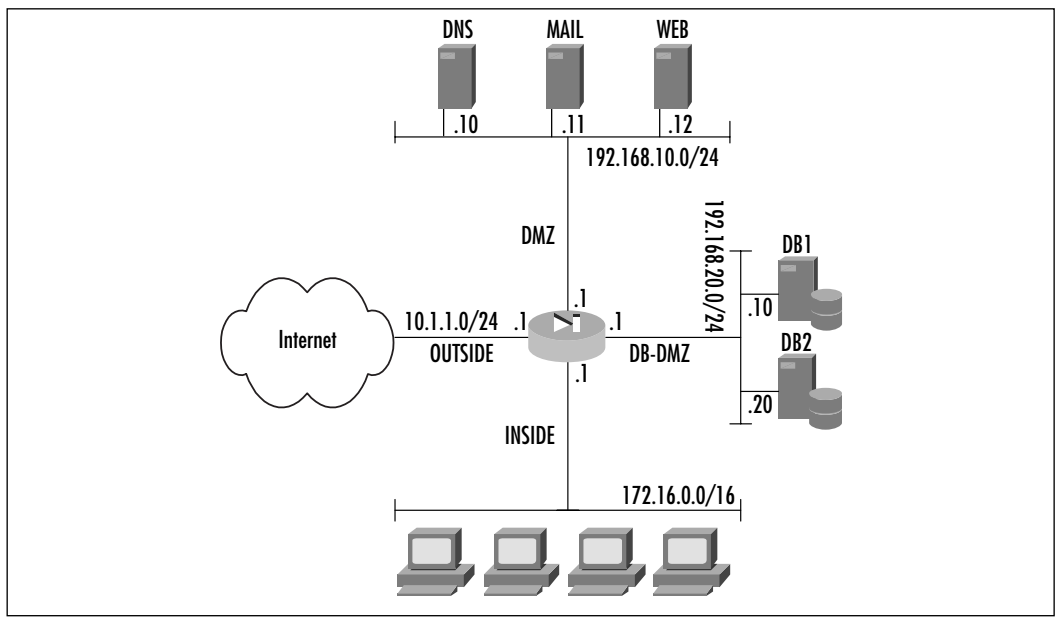
```
PIX1(config)# object-group network all-servers
PIX1(config-network)# group-object net-grp
PIX1(config-network)# network-object 172.16.3.0 255.255.255.0
```

Case Study

We've covered many important topics in this chapter. The following case study will put the concepts and features we learned into action.

Figure 3.6 shows the network layout of the Los Angeles site at Secure Corp. The company has just bought the PIX and needs to configure it. Secure Corp. has already defined a security policy as a precursor to purchasing the PIX. They know how many interfaces they need. The administrators have decided that they need four different security levels to ensure the integrity and security of the network.

Figure 3.6 A Complex Configuration Example



The *inside* interface will be the highest security interface. All corporate users as well as the private and internal servers will be located behind this interface. Private addressing is used for the nodes located behind this interface. The PIX needs to use PAT to translate the IP addresses when the nodes send traffic to the Internet. The PIX should not NAT any traffic from the nodes behind this interface when they access any other interface. There should be no direct access from the Internet to any server located behind this interface. No Internet POP3 and IMAP4 servers are to be available to nodes on the inside network as they are common venues for viruses. All other traffic from the inside network is allowed.

The *db-dmz* interface will have the second highest security level. It is used to host database servers that enable the public Web server to build dynamic HTML pages. No private or confidential information is stored on these database servers. The database servers use private addressing and are the only nodes located behind this interface. The database servers do not need access to the Internet. No direct connections from the Internet should be allowed to the database servers. The database servers are using SQL*Net as the communication protocol to the Web server; therefore they need to be accessible from the Web server on the DMZ interface. The database servers do not need direct access to any hosts on the inside network.

The *dmz* interface will have the third highest security level. Publicly accessible services (Web, mail, and DNS) will be located behind this interface. The servers will use private addressing and require static translations. As these servers may be attacked, access to the Internet and Web should only be allowed from the services that each server provides. Only direct access to the database servers from the Web server on the SQL*Net service is permitted.

The *outside* interface will have the lowest security level. The company wants to only allow access to the services in the DMZ interface. The company also wants to make sure that it will not be the victim of a spoof attack, so it wants to filter out any traffic sourced with a private address. Since the inside network can ping, it is desirable to allow ICMP responses.

We will now discuss the commands to apply this security policy. In the first example, we use only access lists. In the second example, we use conduits and *outbound/apply* statements.

Access Lists

Begin by naming and assigning security levels to the two interfaces not already defined on the PIX:

```
PIX1(config)# nameif ethernet2 dmz security40
PIX1(config)# nameif ethernet3 dbdmz security60
```

Now bring the interfaces online:

```
PIX1(config)# interface ethernet0 auto
PIX1(config)# interface ethernet1 auto
PIX1(config)# interface ethernet2 auto
PIX1(config)# interface ethernet3 auto
```

Assign an IP address to each interface:

```
PIX1(config)# ip address inside 172.16.0.1 255.240.0.0
PIX1(config)# ip address outside 10.1.1.1 255.255.255.0
PIX1(config)# ip address dmz 192.168.10.1 255.255.255.0
PIX1(config)# ip address dbdmz 192.168.20.1 255.255.255.0
```

Assign a default route to the PIX:

```
PIX1(config)# route outside 0.0.0.0 0.0.0.0 10.1.1.254
```

Create access lists to be used later to bypass NAT:

```
PIX1(config)# access-list nonatinside permit ip 172.16.0.0 255.240.0.0
    192.168.10.0 255.255.255.0
PIX1(config)# access-list nonatinside permit ip 172.16.0.0 255.240.0.0
    192.168.20.0 255.255.255.0
PIX1(config)# access-list nonatdbdmz permit ip 192.168.20.0 255.255.255
    .0 192.168.10.0 255.255.255.0
```

Create a global pool utilizing PAT for the inside network:

```
PIX1(config)# global (outside) 1 10.1.1.2
Global 10.1.1.2 will be Port Address Translated
```

Bypass NAT where needed:

```
PIX1(config)# nat (inside) 0 access-list nonatinside
PIX1(config)# nat (dbdmz) 0 access-list nonatdbdmz
```

Enable NAT on the inside interface and have it mapped to the global *id*:

```
PIX1(config)# nat (inside) 1 0 0
```

Create static translations for access from the lower-level security interfaces:

```
PIX1(config)# static (dmz, outside) 10.1.1.10 192.168.10.10
PIX1(config)# static (dmz, outside) 10.1.1.11 192.168.10.11
PIX1(config)# static (dmz, outside) 10.1.1.12 192.168.10.12
PIX1(config)# static (dbdmz, dmz) 192.168.20.0 192.168.20.0 netmask 255
    .255.255.0
```

Configure names for the public addresses of the DMZ servers:

```
PIX1(config)# names
PIX1(config)# name 10.1.1.10 dns
PIX1(config)# name 10.1.1.11 mail
PIX1(config)# name 10.1.1.12 web
```

Configure object groups:

```
PIX1(config)# object-group network dbhosts
PIX1(config-network)# network-object host 192.168.20.10
PIX1(config-network)# network-object host 192.168.20.20
PIX1(config-network)# exit
PIX1(config)# object-group network dmzhosts
PIX1(config-network)# network-object host 192.168.10.1
PIX1(config-network)# network-object host 192.168.10.11
PIX1(config-network)# network-object host 192.168.10.12
PIX1(config-network)# exit
PIX1(config)# object-group icmp-type icmp-outside-in
PIX1(config-icmp-type)# icmp-object echo-reply
PIX1(config-icmp-type)# icmp-object time-exceed
PIX1(config-icmp-type)# icmp-object unreachable
PIX1(config-icmp-type)# exit
```

Configure the access lists for each interface:

```
PIX1(config)# access-list inside_in deny tcp 172.16.0.0 255.240.0.0 any
    eq pop3
PIX1(config)# access-list inside_in deny tcp 172.16.0.0 255.240.0.0 any
    eq 143
PIX1(config)# access-list inside_in permit ip 172.16.0.0 255.240.0.0 any
```

```
PIX1(config)# access-list inside_in permit icmp 172.16.0.0 255.240.0.0 any
PIX1(config)# access-list dbdmz_in permit tcp object-group dbhosts eq
    sqlnet 192.168.10.0 255.255.255.0
PIX1(config)# access-list dbdmz_in permit icmp 192.168.20.0 255.255.255.0
    172.16.0.0 255.255.0.0
PIX1(config)# access-list dbdmz_in deny ip any any
PIX1(config)# access-list dmz_in permit tcp host 192.168.10.11 any eq smtp
PIX1(config)# access-list dmz_in permit tcp host 192.168.10.10 any eq
    domain
PIX1(config)# access-list dmz_in permit udp host 192.168.10.10 any eq
    domain
PIX1(config)# access-list dmz_in permit tcp object-group dmzhosts any eq
    http
PIX1(config)# access-list dmz_in permit tcp host 192.168.10.12 object-
    group dbhosts eq sqlnet
PIX1(config)# access-list dmz_in permit icmp object-group dmzhosts 172.16
    .0.0 255.255.0.0
PIX1(config)# access-list outside_in deny ip 0.0.0.0 255.0.0.0 any
PIX1(config)# access-list outside_in deny ip 10.0.0.0 255.0.0.0 any
PIX1(config)# access-list outside_in deny ip 127.0.0.0 255.0.0.0 any
PIX1(config)# access-list outside_in deny ip 172.16.0.0 255.240.0.0 any
PIX1(config)# access-list outside_in deny ip 192.168.0.0 255.255.0.0 any
PIX1(config)# access-list outside_in deny ip 224.0.0.0 224.0.0.0 any
PIX1(config)# access-list outside_in permit tcp any host web eq http
PIX1(config)# access-list outside_in permit tcp any host mail eq smtp
PIX1(config)# access-list outside_in permit tcp any host dns eq domain
PIX1(config)# access-list outside_in permit udp any host dns eq domain
PIX1(config)# access-list outside_in permit icmp any 10.1.1.0 255.255.255
    .0 object-group icmp-outside-in
PIX1(config)# access-list outside_in deny icmp any 10.1.1.0 255.255.255.0
PIX1(config)# access-list outside_in deny ip any any
```

Apply the access lists to the appropriate interfaces:

```
PIX1(config)# access-group outside_in in interface outside
PIX1(config)# access-group inside_in in interface inside
PIX1(config)# access-group dmz_in in interface dmz
PIX1(config)# access-group dbdmz_in in interface dbdmz
```

Conduits and Outbound/Apply

Name and assign security levels to the two interfaces not already defined on the PIX:

```
PIX1(config)# nameif ethernet2 dmz security40
PIX1(config)# nameif ethernet3 dbdmz security60
```

Bring the interfaces online:

```
PIX1(config)# interface ethernet0 auto
PIX1(config)# interface ethernet1 auto
PIX1(config)# interface ethernet2 auto
PIX1(config)# interface ethernet3 auto
```

Assign an IP address to each interface:

```
PIX1(config)# ip address inside 172.16.0.1 255.240.0.0
PIX1(config)# ip address outside 10.1.1.1 255.255.255.0
PIX1(config)# ip address dmz 192.168.10.1 255.255.255.0
PIX1(config)# ip address dbdmz 192.168.20.1 255.255.255.0
```

Assign a default route to the PIX:

```
PIX1(config)# route outside 0.0.0.0 0.0.0.0 10.1.1.254
```

Create access lists to be used later to bypass NAT:

```
PIX1(config)# access-list nonatinside permit ip 172.16.0.0 255.240.0.0
    192.168.10.0 255.255.255.0
PIX1(config)# access-list nonatinside permit ip 172.16.0.0 255.240.0.0
    192.168.20.0 255.255.255.0
PIX1(config)# access-list nonatdbdmz permit ip 192.168.20.0 255.255.255.0
    192.168.10.0 255.255.255.0
```

Create a global pool utilizing PAT for the inside network:

```
PIX1(config)# global (outside) 1 10.1.1.2
Global 10.1.1.2 will be Port Address Translated
```

Bypass NAT where needed:

```
PIX1(config)# nat (inside) 0 access-list nonatinside
PIX1(config)# nat (dbdmz) 0 access-list nonatdbdmz
```

Enable NAT on the inside interface and have it mapped to the global *id*:

```
PIX1(config)# nat (inside) 1 0 0
```

Create static translations for access from the lower-level security interfaces:

```
PIX1(config)# static (dmz, outside) 10.1.1.10 192.168.10.10
PIX1(config)# static (dmz, outside) 10.1.1.11 192.168.10.11
PIX1(config)# static (dmz, outside) 10.1.1.12 192.168.10.12
PIX1(config)# static (dbdmz, dmz) 192.168.20.0 192.168.20.0 netmask 255
    .255.255.0
```

Configure names for the public addresses of the DMZ servers:

```
PIX1(config)# names
PIX1(config)# name 10.1.1.10 dns
PIX1(config)# name 10.1.1.11 mail
PIX1(config)# name 10.1.1.12 web
```

Configure conduits:

```
PIX1(config)# conduit deny ip any 0.0.0.0 255.0.0.0
PIX1(config)# conduit deny ip any 10.0.0.0 255.0.0.0
PIX1(config)# conduit deny ip any 127.0.0.0 255.0.0.0
PIX1(config)# conduit deny ip any 172.16.0.0 255.240.0.0
PIX1(config)# conduit deny ip any 224.0.0.0 224.0.0.0
PIX1(config)# conduit permit tcp object-group dbhosts eq sqlnet 192.168
    .10.12
PIX1(config)# conduit deny ip any 192.168.0.0 255.255.0.0
PIX1(config)# conduit permit tcp host web eq http any
PIX1(config)# conduit permit tcp host mail eq smtp any
PIX1(config)# conduit permit tcp host dns eq domain any
PIX1(config)# conduit permit udp host dns eq domain any
PIX1(config)# conduit permit icmp 172.16.0.0 255.255.0.0 object-group
    dmzhosts
PIX1(config)# conduit permit icmp 172.16.0.0 255.255.0.0 object-group
    dbhosts
PIX1(config)# conduit permit icmp 10.1.1.0 255.255.255.0 any object-group
    icmp-outside-in
PIX1(config)# conduit deny icmp any any
PIX1(config)# conduit deny ip any any
```

Configure *outbound* statements:

```
PIX1(config)# outbound 10 deny 0 0 0
PIX1(config)# outbound 10 permit 172.16.0.0 255.240.0.0
PIX1(config)# outbound 10 deny 172.16.0.0 255.240.0.0 pop3
PIX1(config)# outbound 10 deny 172.16.0.0 255.240.0.0 143
PIX1(config)# outbound 20 deny 0 0 0
PIX1(config)# outbound 20 except 192.168.10.0 255.255.255.0 sqlnet
PIX1(config)# outbound 30 deny 0 0 0
PIX1(config)# outbound 30 permit 192.168.10.11 255.255.255.255 smtp
PIX1(config)# outbound 30 permit 192.168.10.10 255.255.255.255 domain
PIX1(config)# outbound 30 permit 192.168.10.0 255.255.255.0 http
```

Apply the *outbound* statements to the appropriate interfaces:

```
PIX1(config)# apply (inside) 10 outgoing_src
PIX1(config)# apply (dbdmz) 20 outgoing_src
PIX1(config)# apply (dmz) 30 outgoing_src
```

Summary

Configuring the PIX to pass inbound or outbound traffic requires multiple steps. Basic connectivity allows users on a higher security-level interface of the PIX to transmit traffic to a lower security-level interface using NAT or PAT. This is accomplished using the *nat* command with the *global* command. The PIX ASA allows higher security-level interfaces to transmit traffic to lower security-level interfaces. The PIX is stateful. Users on the inside of the PIX can run almost any application without extra configuration.

Controlling outbound traffic is an important part of a comprehensive security policy. This control can be accomplished using the *access-list* command or the *outbound* command applied to a specific interface. If available, the *access-list* command should be used instead of the *outbound* command to filter traffic. The *access-group* command applies an access list to an interface.

Once outbound access is secure, allowing inbound access is relatively easy. By default, all inbound access (connections from a lower security-level interface to a higher security-level interface) is denied. Access lists or conduits can be used to allow inbound traffic. Conduits are not tied to a particular interface, and the rules defined in a conduit are applied to all inbound traffic. The fundamentals of the *access-list* command are no different for controlling inbound or outbound traffic. For inbound traffic, configuring a static translation (using the *static* command) is required for each publicly accessible server in addition to *access-list* or *conduit*.

Solutions Fast Track

Allowing Outbound Traffic

- ☑ If address translation is configured, the PIX firewall allows all connections from a higher security-level interface to a lower security-level interface.
- ☑ A well-defined security policy usually does not allow all outbound traffic. Define and control what traffic you allow.

- ☑ There are two methods for controlling outbound traffic: access lists and *outbound/apply* statements. Use access lists when possible as they allow greater flexibility. Use the *outbound* and *apply* commands only if you must. These commands are being phased out in newer versions of PIX firewall software.

Allowing Inbound Traffic

- ☑ Connections from a lower security-level interface to a higher security-level interface are denied. To allow inbound traffic, configure a static translation and use access lists or conduits to permit traffic.
- ☑ Port redirection is an excellent option for small businesses that do not have numerous IP addresses.
- ☑ The syntax for access lists is the same whether they are applied to inbound or outbound traffic.

TurboACLs

- ☑ TurboACLs can be enabled for all access lists or on a one-by-one basis.
- ☑ TurboACLs do not speed up access lists of less than 19 lines.
- ☑ TurboACLs do use lots of resources; make sure you have enough available before enabling them.

Object Grouping

- ☑ Object groups simplify access list and conduit configuration and management.
- ☑ There are four types of object groups: ICMP type, network, protocol, and service.
- ☑ Object groups must always be preceded with the *object-group* keyword in an access list or conduit.

Case Study

- ☑ In our case study, the *inside* interface is the highest security interface. All corporate users will be located behind this interface, as well as private and internal servers.
- ☑ The *db-dmz* interface has the second highest security level and is used to host database servers that enable the public Web server to build dynamic HTML pages. No private or confidential information is stored on these database servers.
- ☑ The *dmz* interface has the third highest security level. Publicly accessible services, including Web, mail, and DNS servers, are located behind this interface.
- ☑ The *outside* interface has the lowest security level. The company wants to only allow access to the services in the DMZ interface. The company also wants to make sure that it will not be the victim of a spoof attack, so it wants to filter out any traffic sourced with a private address.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: Could I use a *static* command with a *netmask* option instead of the *nat 0 access-list* command to configure public IP addresses inside the PIX?

A: Although this configuration will work, it opens up the firewall to vulnerabilities if a conduit or access list is misconfigured. Use *nat 0 access-list* if you can.

Q: Why do I have to issue a *clear xlate* after I make changes?

A: The *xlate* table is maintained by the NAT process of the PIX, so if you make changes to that process, items can become stuck in the table or items that should not be in the table might still remain. This can cause unpredictable results, and it creates a security risk.

Q: Should I move all my servers into a DMZ?

A: DMZs are very helpful in containing security risks for publicly accessible servers. If a server is not needed by the outside world, there is probably no reason to move it into a DMZ. If you do not trust your inside users, that is another story.

Q: Why should I use private IP addresses inside my network if I have enough public address space?

A: Using private address space inside your network has many advantages. The amount of address space provided allows for large flexibility in the network design and allows for expansion. However, private addresses are not for everyone, and many universities and other institutions that have large amounts of IP address space use public addressing in their networks.

Q: How do I know if my access lists are working correctly?

A: The *show access-list* command displays the current access list configuration on the PIX. If you want to know that the access lists are working, watch the *hitcnt* counter. Every time traffic matches an entry, the counter will increment.

Chapter 4

Advanced PIX Configurations

Solutions in this chapter:

- **Handling Advanced Protocols**
- **Filtering Web Traffic**
- **Configuring Intrusion Detection**
- **DHCP Functionality**
- **Other Advanced Features**

☑ **Summary**

☑ **Solutions Fast Track**

☑ **Frequently Asked Questions**

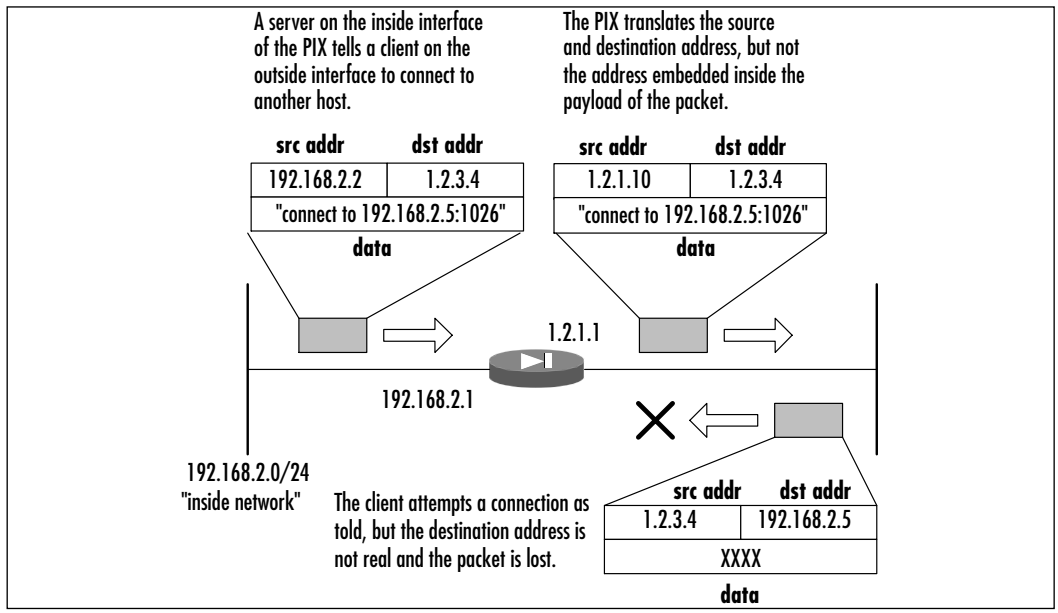
Introduction

Now that you have learned how to pass simple traffic through the PIX firewall, we are ready to dive in and deal with configurations that are more complex. In this chapter, we discuss some of the more advanced features that the PIX firewall has to offer. You will learn how the PIX can be configured to handle complex protocols that operate over multiple or dynamic ports. In some cases, these protocols embed IP addresses and port information inside the payload of data packets, creating a challenge for performing NAT/PAT. The PIX firewall also has the ability to block Web traffic, including Java and ActiveX applications. The PIX firewall provides integrated intrusion detection features for common information-gathering stacks and network attacks. We will look at how to use the integrated IDS signature in the PIX firewall to detect patterns of network misuse. In small office/home office (SOHO) environments, it might be beneficial to use the DHCP client and server functionality provided by the PIX firewall. In this chapter we examine both of these features in detail and show how to use them. Finally, we complete this chapter by discussing unicast and multicast routing, PPPoE, and reverse-path forwarding.

Handling Advanced Protocols

One of the most important features of all firewalls is their ability to intelligently handle many different protocols and applications. If all our needs were satisfied by devices that simply allow, say, outgoing connections to port 80 (HTTP) and deny incoming connections to port 139 (NetBIOS), the life of a security engineer would be much simpler. Unfortunately, many applications, some of which were developed even before the idea of a firewall emerged, act in a much more complicated manner than Telnet or HTTP. One of earliest examples is File Transfer Protocol, or FTP (which we discuss in detail in the next section). The general problem these applications pose is that they use more than one connection to operate and only one of these connections occurs on a well-known port, while the others use dynamically assigned port numbers, which are negotiated in the process of communication. Figure 4.1 shows an example of what happens when this situation occurs and no special measures are in place. (This is a simplified example of SQL*net session negotiation.)

Figure 4.1 Client Redirection Without Application Inspection



Thus, any firewall that wants to handle these negotiations well needs the ability to monitor them, understand them, and adjust its rules accordingly. This situation becomes even more complicated when NAT or PAT are involved; the firewall might need to change the data portion of a packet that carries embedded address information in order for the packet to be correctly processed by a client or server on the other side of PIX. There are many implementations of this feature for various firewalls—for example, Stateful Inspection in the Check Point product family or the Adaptive Security Algorithm (ASA) of Cisco PIX devices.

The ASA uses several sources of information during its operation:

- Access control lists (ACLs), which allow or deny traffic based on hosts, networks, and the TCP or UDP ports involved.
- Internal translation (xlate) and connection (xlate) tables, which store information about the state of the established connections and are used for fast processing of the traffic that belongs to these connections.
- Embedded rules for application inspection, which allow automatic processing of most of the complicated cases mentioned. Although some of these rules are configurable, others are fixed.

A detailed description of ASA was provided in Chapter 3. Here we look at the processing of a TCP packet by ASA, including application-level intelligence (not considering address translation):

1. If the packet is not the first one in a connection (with the SYN bit set), it is checked against internal tables to decide if it is a reply to an established connection. If it is not, the packet is denied.
2. If it is a SYN packet, it is checked against internal tables to decide if it is a part of another established connection. If it is, the packet is permitted and internal tables are adjusted in order to permit return traffic for this connection.
3. If this SYN packet is not a part of any established communication, it is checked against ACLs.
4. If the SYN packet is permitted, the PIX creates a new entry in internal tables (the XLAT and/or CONN table).
5. The firewall checks to see whether the packet needs additional processing by application-level inspection algorithms. During this phase, the firewall can create additional entries in internal tables. For example, it can open a temporary conduit for an incoming FTP connection based on the *PORT* command that it sees in the packet. "Temporary" means that this conduit will exist only until the FTP session terminates and will be deleted after the session is closed.
6. The inspected packet is forwarded to the destination.

The situation for UDP is similar, although simpler because there are no distinct initial packets in the UDP protocol, so the inspection simply goes through internal tables and ACLs and then through application inspection for each packet received. Figure 4.2 illustrates how the same example from Figure 4.1 would work with application inspection turned on.

The PIX uses source/destination port numbers to decide if application inspection is needed for a particular packet. Some of these ports are configurable and others are not. Table 4.1 summarizes the application inspection functions provided by PIX firewall software version 6.2.

Figure 4.2 Application Inspection in Action

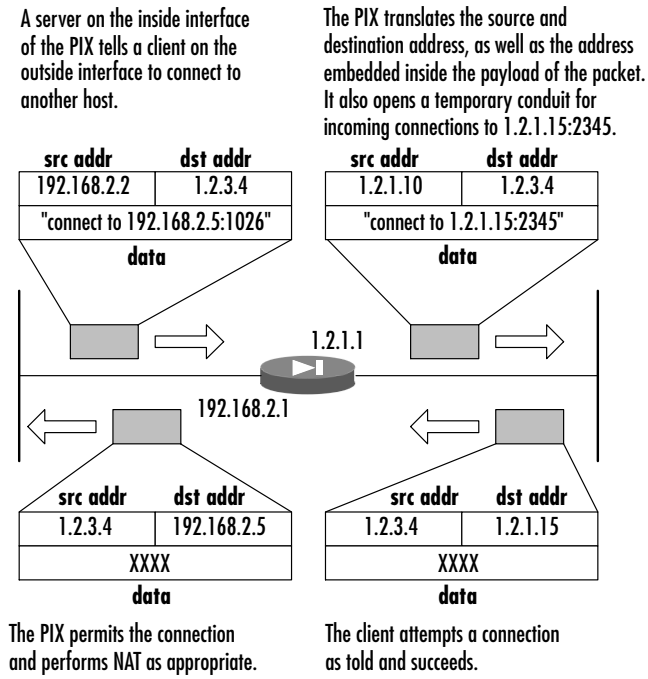


Table 4.1 Application Inspection Features of Cisco PIX Firewall Version 6.2

Application	PAT Support	NAT 1-1 Support	Config-urable?	Default Port	Related Standards
H.323	Yes	Yes	Yes No	TCP/1720 UDP/1718	H.323, H.245, H.225.0, Q.931, Q.932
H.323 RAS	Yes	Yes	Yes	UDP/1719	N/A
SIP	Yes	Yes	Yes No	TCP/5060 UDP/5060	RFC 2543
FTP	Yes	Yes	Yes	TCP/21	RFC 1123
LDAP (ILS)	Yes	No outside NAT	Yes	TCP/389	N/A
SMTP	Yes	Yes	Yes	TCP/25	RFC 821, 1123
SQL*Net v.1, v.2	Yes	Yes	Yes	TCP/1521 (v.1)	N/A

Continued

Table 4.1 Continued

Application	PAT Support	NAT 1-1 Support	Config-urable?	Default Port	Related Standards
HTTP	Yes	Yes	Yes	TCP/80	RFC 2616
RSH	Yes	Yes	Yes	TCP/514	Berkeley UNIX
SCCP	No	Yes	Yes	TCP/2000	N/A
DNS	Yes	Yes	No	UDP/53	RFC 1123
NetBIOS over IP	See next two entries				
NBNS/UDP	No	No	No	UDP/137	N/A
NBDS/UDP	Yes	Yes	No	UDP/138	N/A
Sun RPC	No	No	No	UDP/111 TCP/111	N/A
XDCMP	No	No	No	UDP/117	N/A
RTSP	No	No	Yes	TCP/554	RFC 2326, 2327, 1889
CU-SeeMe	No	No	No	UDP/7648	N/A
ICMP	Yes	Yes	No	N/A	N/A
VDO Live	No	Yes	No	TCP/7000	N/A
Windows Media (NetShow)	No	Yes	No	TCP/1755	N/A

The main command that is used to configure the services stated as "configurable" in Table 4.1 (FTP, H.323, HTTP, ILS, RSH, RTSP, SIP, SSCP, SMTP, and SQL*Net) is the *fixup* command. Its basic syntax is:

```
[no] fixup protocol [protocol] [port]
```

The following sections describe how this command is used for each protocol. Depending on the protocol it is used with, application inspection (fixup) provides the following functionality for complex protocols:

- Securely and dynamically open and close temporary conduits for legitimate traffic
- Network Address Translation
- Port Address Translation
- Inspect traffic for malicious behavior

File Transfer Protocol

One of the first application-level protocols that posed problems for simple packet-filtering devices was FTP, which is documented in RFC 959. FTP always uses two connections for operation. The first one, known as the *control connection*, is a connection from the client FTP program to the server's FTP port (TCP port 21 by default). This connection is used for sending commands to the server and receiving informational replies. These commands and replies are a little different from what you enter on the keyboard. For example, when you log into an FTP server and enter your username, your FTP client sends the *USER username* command to the server and probably receives a reply *331 User name okay, need password*. It then asks you for your password, and the login process completes.

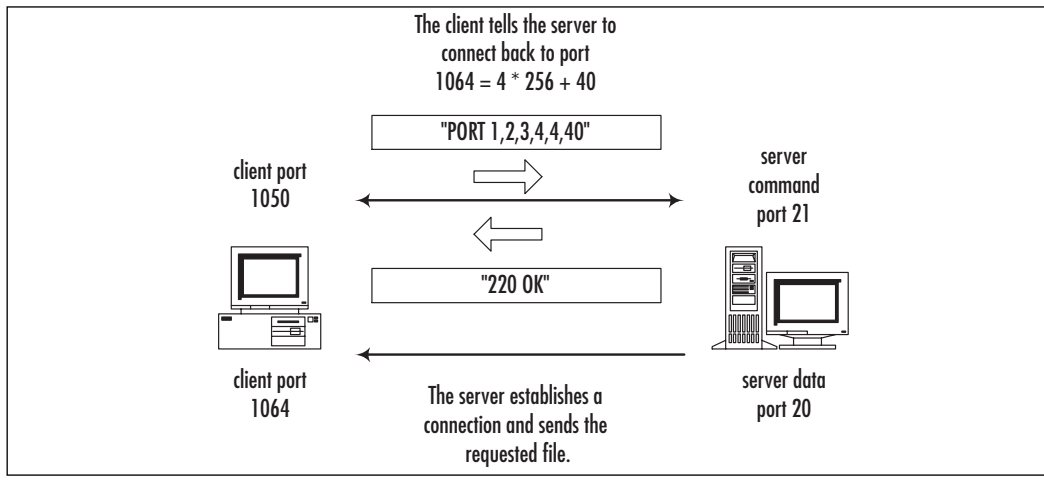
The second connection is opened for the actual file transfer operation and can behave differently depending on the mode in which the client is operating; it can be initiated either by the client or by the server. The main difference is whether the client tells the server to operate in *passive* or *active* mode.

Active vs. Passive Mode

The first FTP servers and clients used *active* mode, where a file transfer happens as shown in Figure 4.3 and described here:

1. When the client (already connected to the server's FTP control port and logged in) needs to receive a file from the server, it sends a *PORT A1,A2,A3,A4,a1,a2* command, where A1, A2, A3, and A4 are the four octets of the client's IP address and *a1* and *a2* are the port numbers on which it will listen for connections. This port number is an arbitrary value and is calculated as a1*256+a2.
2. After receiving a *200 OK* reply from the server, the client sends the *RETR* command to start the transfer.
3. The server opens a connection to the port that the client specified and pipes the file's contents into this connection. After the file is transferred, this data connection is closed, while the control connection stays open until the client disconnects from the server. The source port of this connection is "ftp-data," TCP port 20.

Figure 4.3 Active FTP Connection Flow



Now, if the client is behind a firewall (or, in PIX terms, is on a higher security-level interface than the server), the connection from the server is likely to be refused unless the firewall permits inbound connections to all high ports on the client side, which is of course not good. The PIX firewall can monitor FTP control connections, so when it discovers a *PORT* command issued by the client, it temporarily permits inbound connections to the port requested by the client in this command.

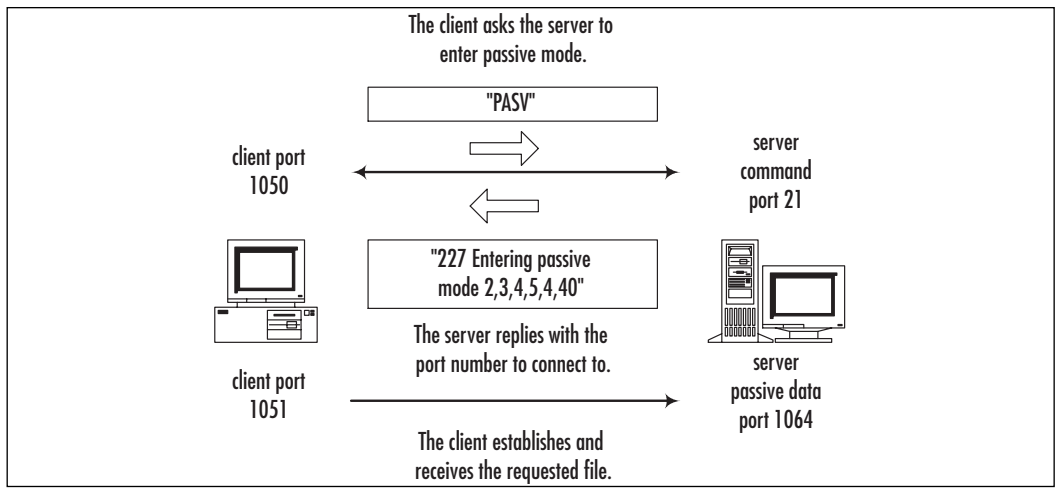
The other issue here is that when NAT or PAT are used, the PIX also translates the address and port number (A1.A2.A3.A4:a1a2) inside this command to the NATted IP and port. For example, if the client's address is 10.0.0.1 and it is translated to 1.2.3.4, the *PORT 10,0,0,1,4,10* command the client issued (which says that the client is ready to receive connections to 10.0.0.1:1034) during its transit through the PIX will be translated to something like *PORT 1,2,3,4,8,10*, so that the server will open the data connection to 1.2.3.4:2058. This destination will be properly translated by the PIX to 10.0.0.1:1034 using its internal tables.

The second mode of FTP operation is *passive* mode. In this mode, a file transfer happens as shown in Figure 4.4 and described here:

1. Soon after connecting to the server's FTP control port and logging in, the client sends the *PASV* command, requesting the server to enter the passive mode of operation.
2. The server responds with "227 Entering Passive Mode A1,A2,A3,A4,a1, a2." This response means that the server is now listening for data connections on the IP address and port it has specified in the reply.

3. The client connects to the specified port and sends the *RETR* command to start the transfer.
4. The server sends the file's contents over this second (data) connection.

Figure 4.4 Passive FTP Connection Flow



This mode of operation does not cause a problem when the client is on a more secure interface, since by default the client is permitted to initiate any outbound connections. Unfortunately, there is a problem when the server is on a more secure interface than the client; the firewall will generally not allow the client to open an inbound connection on an arbitrary port. To overcome this problem, the PIX firewall monitors *PASV* commands and "227" replies, temporarily permits an inbound connection to the specified port, and modifies IP addresses and port numbers to correspond with NATted ones.

The described behavior of the PIX firewall is turned on by default; it inspects inbound and outbound connections to FTP control port 21. To turn it off or modify the port numbers on which it should perform inspection, use the *fixup protocol ftp* command in configuration mode. The syntax of this command is as follows:

```
[no] fixup protocol ftp [strict] [<port>]
```

Here, *port* is the port number used for control connections, *PORT* commands, and "227" replies. The default state of FTP inspection is equal to:

```
fixup protocol ftp 21
```

If you enter extra *fixup* commands, the ports specified in them are inspected simultaneously for incoming and outgoing FTP control connections. For example, if you enter *fixup protocol ftp 2100*, both default the default port (21) as well as port 2100 will be inspected. The command *no fixup protocol ftp [port]* disables the previously entered *fixup* command. For example, to enable processing of only connections to port 2100, you need to configure the following:

```
PIX1(config)# fixup protocol ftp 2100
PIX1(config)# no fixup protocol ftp 21
```

It is possible to disable inspection of FTP connections using:

```
no fixup protocol ftp
```

The result will be that inside users are able to initiate FTP connections to outside hosts only in passive mode, not active mode. Outside clients will be able to initiate FTP connections to inside servers in active mode only (assuming there is a static NAT entry and an access list or conduit in place), not passive mode. To reset application inspection to the standard port settings for all protocols at the same time, use the *clear fixup* command.

The full functionality of FTP application inspection consists of the following tasks:

1. Tracking of FTP command and response sequence (*PORT* and *PASV* commands and "227" replies).
2. Creating a temporary conduit for the data connections based on the result of this tracking (if necessary).
3. NATting of IP addresses inside the commands and replies.
4. Generating an audit trail.

An audit trail is generated in the following cases:

- An audit record 302002 is generated for each uploaded or downloaded file.
- Each download (*RETR*) or upload (*STOR*) command is logged.
- File operations are logged together with the FTP username, source and destination IP addresses, and NAT address.
- An audit record 201005 is generated if the firewall failed to allocate a secondary channel due to memory shortage.

In the first implementations of FTP inspection, the process of looking for the relevant commands/replies in IP packets was very simple: The PIX only looked for a string such as *PORT* inside the packet and tried to interpret it as a corresponding command. Of course, various attacks were designed to fool the firewall into opening an extra port by sending bogus commands and replies from the client or the server (see www.cisco.com/warp/public/707/pixftp-pub.shtml).

Since then, the inspection process has been greatly improved, and another option, *strict*, has been introduced to perform much more rigorous checks on the command/response stream. If you use this option in configuration of FTP inspection—for example, *fixup protocol ftp strict 21*—the firewall imposes much more rigorous restrictions on the command/response flow. These restrictions can sometimes break applications that are not fully RFC compliant. If one of the following problems is encountered, the connection is denied or dropped:

- Clients are prevented from sending embedded commands. The connection that tries to use these commands is closed. This action is performed by checking how many characters are present in the *PORT* or *PASV* command after the IP address and port number. If there are more than eight characters, it is assumed that it is an attempt to add another command at the end of the line, and the connection is dropped.
- Before a new command is allowed, the server should send a reply to each command received.
- Only servers can generate "227" messages (protection against reply spoofing) and only clients can generate *PASV* and *PORT* commands (protection against command spoofing). The reason here is that without *strict*, a client can send any garbage to the server, including fake "227" messages—for example, *227 foobar A1, A2, A3, A4, a1, a2,* and although the server replies with an error message, the firewall could be fooled into permitting the connection with the parameters specified.
- Extra checking of "227" and *PORT* commands is performed to ensure that they are really commands/replies, not a part of some error message.
- Truncated commands; *PORT* and *PASV* commands are checked for the correct number of commas in them. Each should contain only five commas (see previous examples).
- Size of *RETR* and *STORE* commands; their length (including the filename for download/upload) should not be greater than an embedded constant. This is done to provide protection against possible buffer overflows.

- Invalid port negotiation; the port number used for the data connection must be a high port (that is, a port with number greater than 1024).
- Every FTP command sent by the client must end with *<cr><lf>* characters, as specified by RFC 959.

Domain Name Service

The main task of application inspection for DNS (known as *DNS Guard*) is to impose specific restrictions on DNS requests over UDP that pass through the firewall (compared with the generic processing of all UDP communications). Roughly speaking, the data part of each DNS request contains a serial number (ID) and the body of the request. For example, requests for "A-records" (address records) include the DNS name for which an IP address is sought. The reply to this request should contain the same ID and an IP address.

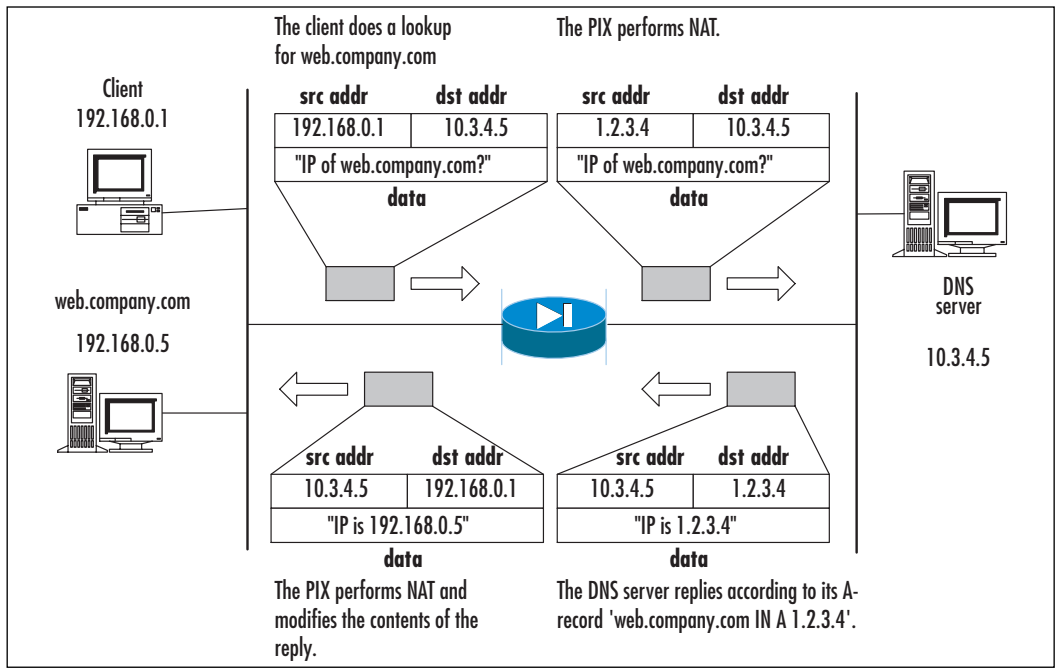
DNS Guard ensures the following:

- Only replies with the correct ID are accepted.
- Only one reply is accepted. In the case of multiple replies, all but the first one are ignored.
- The UDP connection associated with the DNS connection is destroyed as soon as a DNS reply is received, not after the UDP timeout has expired.
- IP addresses in A-record replies are translated if necessary. This process is controlled by the *alias* command. It also translates addresses to be consistent with NAT statements, including outside NAT, which was introduced in version 6.2. Generally, the *alias* command is not needed because of this outside NAT feature.

As an example for the last case, consider the configuration in which a client (192.168.0.1) and a Web server (web.company.com, IP address 192.168.0.5) are located on the inside interface of PIX and have nonroutable addresses. A DNS server is on the outside. The PIX is configured to translate both the client and the server addresses via PAT to a single IP of 1.2.3.4. This address is recorded on the DNS server as an address for web.company.com. When a client requests an IP address (an A-record) for the server, the PIX forwards the request to the DNS server, translating the source IP. When it receives the DNS server's reply, it not only translates the packet's destination IP address (changing 1.2.3.4 to

192.168.0.1), but it also changes the address of the Web server contained in the reply's data field (that is, 1.2.3.4 contained in the reply is changed to 192.168.0.5). As a result, the internal client will use the internal address 192.168.0.5 of the Web server to directly connect to it. Figure 4.5 illustrates how the DNS request and reply pass through the PIX.

Figure 4.5 The DNS Guard Operation



When the DNS server is on a more secure interface than the Web server and/or client, either outside NAT (preferred in version 6.2) or *alias* commands are used. Outside NAT is very similar to the previous situation. Before version 6.2, you needed to use the *alias* command *alias internal_server_address external_server_address* in order to process A-record replies properly in this case.

NOTE

When using *alias* commands for DNS fixups, you need to turn off proxy ARP on the internal interface, using the *sysopt noproxyarp inside_interface* command. It is also possible to turn off processing of DNS replies for addresses stated in the *alias* commands by using the *sysopt nodnsalias* command.

It is not possible to disable application inspection of DNS or change the DNS port from the default of 53.

Simple Mail Transfer Protocol

Similar to FTP and DNS inspection, application inspection of Simple Mail Transfer Protocol (SMTP), also known as *Mail Guard*, is designed to restrict what servers and clients can do and see while not harming the essential functionality of the protocol—sending electronic mail.

SMTP is described in RFC 821 as a Telnet-based protocol designed for transferring electronic mail between servers. The client sends commands to the server, and the server replies with status messages and probably some extra information. In essence, it is very simple: There are commands for specifying a recipient of the message, the sender, and the message itself. An example of an SMTP session is shown in Figure 4.6.

Figure 4.6 An SMTP Session

```
Server: 220 Simple Mail Transfer Service Ready
Client: HELO example1.com
Server: 250 OK
Client: MAIL FROM:<Alice@example1.com>
Server: 250 OK
Client: RCPT TO:<Bob@example2.com>
Server: 250 OK
Client: RCPT TO:<John@example2.com>
Server: 550 No such user here
Client: DATA
Server: 354 Start mail input; end with <CRLF>.<CRLF>
Client: Blah blah blah...
Client: ...foobar.
Client: <CRLF>.<CRLF>
Server: 250 OK
Client: QUIT
Server: 250 OK
```

This transcript shows the session in which the client tried to send e-mail from Alice@example1.com to Bob@example2.com, which was accepted, and to John@example2.com, which was rejected because a user was not found.

These commands (*HELO, MAIL, RCPT, DATA,* and *QUIT*), together with a couple of control commands (*NOOP*, do nothing, and *RSET*, reset state) make up a minimal set required by RFC 821, section 4.5.1.

Mail Guard is turned on by default on port 25 and can be reconfigured using the following command:

```
[no] fixup protocol smtp [<port>[-<port>]]
```

This command functions in the same way as *fixup protocol ftp* except that it is possible to specify a range of TCP ports instead of only one.

WARNING

When enforcing a minimal command set, the PIX causes some problems with Microsoft Exchange servers and Outlook clients. The problem here is that Microsoft's implementation of SMTP is not strictly RFC 821 compliant and uses the *EHLO* command instead of *HELO* to start a connection. The PIX changes this command to *NOOP*, so the server simply returns a "250 OK" reply, which is interpreted as a confirmation that the server supports SMTP extensions. Consequently, clients do not fall back to the *HELO* command and continue using extended features (see RFC 2821), which are blocked by the PIX. Most non-Microsoft clients, though, after receiving a simple "250 OK" reply instead of a more informative *EHLO* response, do fall back to the *HELO* style of operations and everything works well.

The main goal of Mail Guard is to restrict commands clients use to the minimal set described, while monitoring the entire command/response sequence and generating a specific audit trail. In detail:

- Mail Guard monitors commands sent by a client, and if a command does not belong to the minimal set, it is replaced with the *NOOP* command.
- If Mail Guard encounters an unknown command, the whole data portion of a TCP/IP packet is filled with the *X* symbol, which, when received by a server, causes the server to produce an error.
- *MAIL* and *RCPT* commands are monitored for correct usage of <, >, and | characters. The pipeline character | is replaced with a space, and < and > are allowed only when they appear as delimiters of an e-mail address. When an invalid character is replaced in the e-mail address, audit record 108002 is generated.

- Mail Guard checks for truncated or incorrectly terminated commands (ones that do not end with *<cr><lf>*).
- In a banner message—for example, "220 foobar email server ready"—all symbols except "220" are changed to *X*. This is done in order to hide details about the server platform or operating system, which are often reported in these banners.

Hypertext Transfer Protocol

With HTTP application inspection active, all traffic to and from the specified ports is subject to the following:

- Logging of all HTTP GET requests
- Screening of URLs by either a Websense or an N2H2 server
- Filtering of ActiveX and Java content

The command for using application inspection for HTTP is shown here:

```
[no] fixup protocol http [<port>[-<port>]]
```

As with SMTP, it is possible to state a range of ports. The default port is 80. URL screening and active content filtering are described later in the chapter, in the "Filtering Web Traffic" section, and is configured using the *filter* command. Note that when you turn HTTP inspection off using *no fixup protocol http*, all HTTP inspection is disabled, even if URL screening rules are configured.

Remote Shell

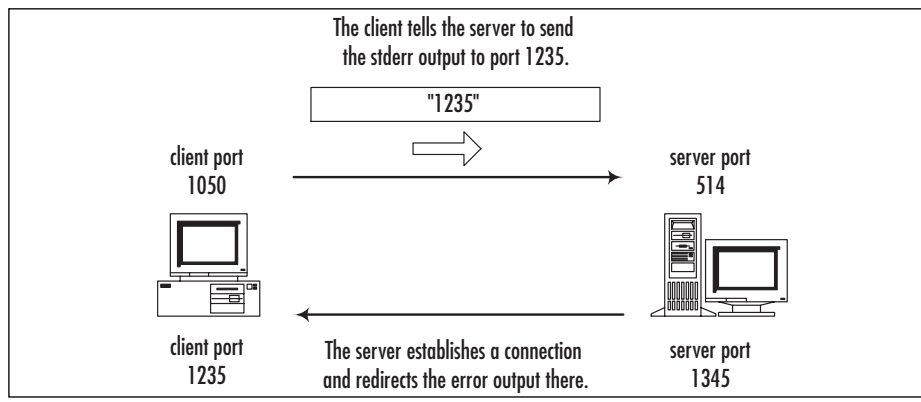
The *r-utilities* (*rsh, rcp, rexec,* and *rlogin*) were developed to be convenient tools for remote command executions on UNIX machines, without the need for logging in as in Telnet. These utilities are inherently very insecure and are being phased out everywhere and replaced by SSH-based tools. Probably the only important application that still uses these utilities is CVS, although it is also being changed to use SSH-based means of authentication and file transfer.

Having said that, let's consider how this protocol works and why it poses problems for firewalls. When you try to connect to a remote host via Remote Shell (rsh), the following happens:

1. The rshd server on the remote host listens on a specified port (TCP port 514, by default) for incoming connections. The client establishes a connection to this port.

2. Immediately after the connection is established, the client sends an ASCII-coded number to the server. This is the port number that the server should use for establishing a secondary connection back to the client. This secondary connection is established so that the server can send any error output to the client. (More precisely, the server will pipe a stderr stream to this secondary connection.) This port number is not fixed, so if the firewall does not allow arbitrary connections to the client—for example, when the client is on a more secure interface)—this secondary connection from the server to the client will fail. In this case, the server closes the first connection and generates an error message, "Can't make pipe." See Figure 4.7 for an example of connection flow.
3. After an inbound connection to the client is established, the server performs client authentication. The client sends the server a command to be run on the server and receives the results of its execution (stdout stream) on the first connection, plus any errors that occurred on the second connection.
4. Both connections are closed.

Figure 4.7 RSH Connection Establishment



In order to process outbound rsh connections, the PIX monitors the initial connection, notes the port number the client requested, and opens a temporary conduit for the incoming connection by the server. The PIX is also able to perform PAT for this port if it is needed. The command to enable or disable application inspection for rsh is:

```
[no] fixup protocol rsh <port>
```

Inbound rsh connections do not need any special processing, only an access-list entry or conduit for an outside client to reach port 514 (default port for rsh) on the inside server.

Remote Procedure Call

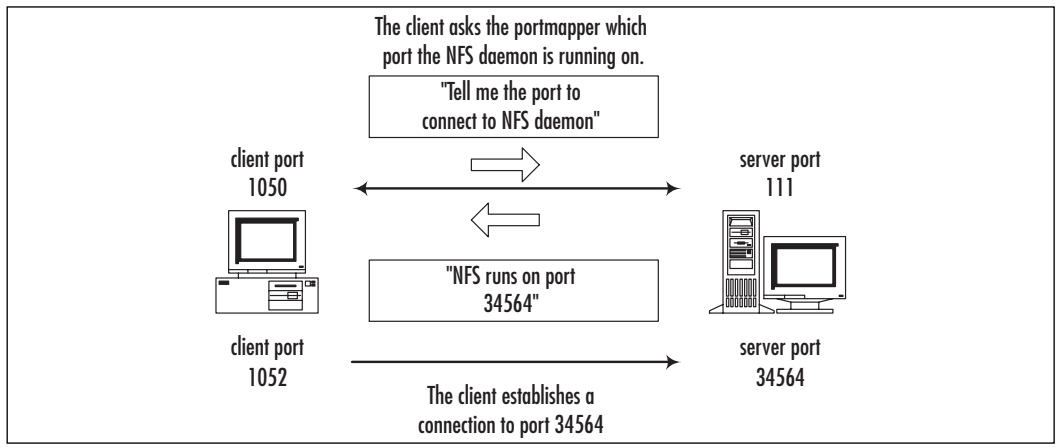
Remote procedure call (RPC) is a very general mechanism for client-server applications developed by Sun Microsystems. Many applications are built on top of this system, the most important of which are Network File System (NFS) and Network Information System (NIS), which are used in many UNIX networks.

The RPC server is a collection of procedures, each of which can be called by a client sending an RPC request to the server, possibly passing some parameters. The server runs the required procedure and sends the results to the client. This data exchange is platform-independent and is encoded using External Data Representation (XDR) format. Each procedure is identified by an assigned program number, which the client indicates in the request. The default correspondence between program numbers and procedures is stored on UNIX hosts in the /etc/rpc file. To further complicate things, an RPC server can run various versions of each program at the same time. In this case, the version numbers are added to the request.

On TCP/IP networks, each version of a program running on the server is assigned a TCP and a UDP port (both ports have the same number). In order for this service to be generic (and because RPC programs do not use reserved port numbers), there is no fixed correspondence between program names (or numbers) and the ports they are running on. The ports are assigned dynamically by a separate daemon called *portmapper*, which functions as a multiplexing service. Each program has to register with portmapper in order to be available for RPC calls. Portmapper then reserves a TCP and a UDP port for it. When a client wants to make a call to a remote procedure, it first queries the portmapper daemon (which runs on port 111 by default), sending it a program number and receiving the number of a port it runs on. The client then connects to this port and interacts directly with the required program. Figure 4.8 illustrates this process.

Here, the problem for a firewall arises when the RPC server is on a more secure interface; it is simple to set up a conduit permitting incoming connections to the portmapper port 111, but it is not possible to know beforehand which extra ports need to be opened for incoming RPC requests to specific programs. The PIX does the following:

Figure 4.8 RPC Connection Flow



1. It inspects all outgoing packets that have a source port of 111.
2. When it notices a portmapper reply with some port number, the PIX opens embryonic TCP and UDP connections on this port.
3. The PIX does not inspect RPC packets for anything else. For example, it does not attempt to translate embedded IP addresses.

This feature is not configurable.

Real-Time Streaming Protocol, NetShow, and VDO Live

In this section, we examine streaming applications and the problems they pose to firewalls. Streaming is a form of communication in which the client requests that the server send data at a certain speed. In some implementations, the client needs to confirm each portion of data received. In others, the server just sends data until the client tells it to stop. Major protocols widely used in this area are Real-Time Streaming Protocol, or RTSP (used by RealPlayer, Cisco IP/TV, and Apple QuickTime 4), NetShow (used by Microsoft Media Player), and VDO Live.

The RTSP, defined in RFC 2326, is used for session setup and teardown as well as for controlling data flow (stop, play, pause). The RFC allows RTSP to run over both TCP and UDP, but all commercial implementations use only TCP, so Cisco supports application inspection for TCP-based RTSP sessions only. RTSP is a text-based, HTTP-like protocol by which the client sends requests and obtains replies from the server. Requests may be used to negotiate the transport

that will be used for streaming data transmission, the options that are supported, asking the server to start or stop streaming, and the like. Embedded in RTSP is Session Description Protocol (SDP, described in RFC 2327), which is used to provide the client with some extra information about the source of a datastream, including its physical location (in terms of IP addresses). The following is an example of an RTSP/SDP session (with nonrelevant parts skipped):

```
C> OPTIONS rtsp://www.play.com:554 RTSP/1.0
C> CSeq: 1
S> RTSP/1.0 200 OK
S> CSeq: 1
S> Server: RealMedia Server Version 6.0.3.354 (win32
S> Public: OPTIONS, DESCRIBE, ANNOUNCE, SETUP, GET_PARAMETER,
    SET_PARAMETER, TEARDOWN
S> RealChallenge1: 15d67d72b49fd4895774cfbb585af460
<skipped>
C> SETUP rtsp://www.play.com:554/g2audio.rm/streamid=0 RTSP/1.0
C> CSeq: 3
C> RealChallenge2: 319cd1020892093a7b7290ef22b6f41101d0a8e3, sd=3d00792f
C> Transport: x-real-rdt/mcast;client_port=6970;mode=play,x-real-
    dt/udp;client_port=6970;mode=play,x-pn-tng/udp;client_port=6970;
        mode=play,rtp/avp;unicast;client_port=6970-6971;
            mode=play
S> RTSP/1.0 200 OK
S> CSeq: 3
S> Session: 22660-2
S> RealChallenge3: 9521b5d0fcff7ab0ea7f407f89c5f3584f213d09,sdr=9bf7e48f
S> Transport: x-real-rdt/udp;client_port=6970;server_port=28344
<skipped>
C> PLAY rtsp://www.play.com:554/g2audio.rm RTSP/1.0
C> CSeq: 5
C> Session: 22660-2
S> RTSP/1.0 200 OK
S> CSeq: 5
S> Session: 22660-2
C> TEARDOWN rtsp://www.play.com:554/g2audio.rm RTSP/1.0
C> CSeq: 6
C> Session: 22660-2
```

```
S> RTSP/1.0 200 OK
S> CSeq: 6
S> Session: 22660-2
```

The session starts by negotiating client and server capabilities. Then comes the *SETUP* command, in which the transport mode (RDT or RTP) and port are negotiated (highlighted in italics in the preceding code). The client then commands the server to start transmission, and it finally tears the connection down after all data has been received.

Real Data Transport (RDT) is a RealNetworks proprietary protocol for data delivery. It uses two one-way UDP connections: one from the server to the client for data delivery and another from the client to the server for requests to retransmit lost packets. This is the default mode for the RealNetworks G2 server. In the exchange that appears in the preceding code, the client has chosen to receive data on port 6970 and the server has chosen to receive requests on port 28334.

Real-Time Transport Protocol (RTP), described in RFC 1889, uses a one-way UDP connection for sending data from the server to the client and another two-way UDP connection for transmission control with RTP Control Protocol (RTCP). RTP/RTCP connections occur on two consecutive ports: the RTP channel is an even number port and RTCP is the next consecutive port. This is the default mode for Apple QuickTime and Cisco IP/TV.

To further complicate matters, there is one more mode of operation, interleaved mode, in which all RDT and RTP communications are embedded into the initial RTSP connection. This is the simplest mode from the firewall's point of view because it requires no extra processing.

RTSP connections occur on the default port of 554. Cisco IP/TV also uses port 8554, which is not enabled by default on the PIX. The command for enabling and disabling RTSP inspection is:

```
[no] fixup protocol rtsp [<port>]
```

For example, in order to enable correct processing of Cisco IP/TV streams, you need to add the following command to the default configuration:

```
PIX1(config)# fixup protocol rtsp 8554
```

When they perform application inspection for the RTSP protocol, the PIX monitors all *SETUP* replies with a code of "200." If the message is inbound and the server is a less secure interface, the firewall needs to open a temporary conduit for the incoming connection from the server to the client on a port stated in

the reply. If the message is outbound, no extra actions are needed. The inspection process has the following restrictions:

- The PIX monitors only TCP-based RTSP exchange. RTSP over UDP is not inspected.
- RealNetworks RDT multicast mode is not supported (x-real-rdt/mcast content type).
- Proprietary RealNetworks PNA mode is not supported.
- The PIX is unable to recognize RTSP embedded in HTTP.
- RealPlayer needs to be set up to use only TCP to connect to the server (that is, to use RTSP over TCP only). This is done via Options | Preferences | Transport | RTSP Settings. The relevant setting here is Use TCP to Connect to Server. You can further configure it to work in interleaved mode (which needs no application inspection) by selecting **Attempt to use TCP for all content**. You can also configure it to use RDP by selecting **Attempt to use UDP for all content**.
- Supported RDP transports are rtp/avp, rtp/avp/udp, x-real-rdt, x-real-rdt/udp, and x-pn-tng/udp.

Even if the PIX tries its best to fix addresses inside RTSP/SDP packets, many NAT/PAT restrictions apply:

- PAT is not supported.
- NAT of SDP messages inside RTSP is not supported because these long messages could be split into several packets and the firewall has no means of reconstructing the original message. On the other hand, NAT usually works with Cisco IP/TV RTSP messages.
- NAT of datastream-related connections can be performed for RealNetworks server and Apple QuickTime. For Cisco IP/TV it can only be done when the viewer and the content manager are on the outside interface and the server is on the inside.

Microsoft's NetShow, used by Media Player, is a less complex streaming protocol. Like the other streaming protocols, it has a control channel, which is used to negotiate setup and teardown of a data delivery channel. The data channel can be either TCP- or UDP-based. When UDP streams are used, the following process occurs:

1. The client connects to the server on TCP port 1755.
2. After a connection is established, the client sends a message to the server, proposing a UDP port on which it is going to receive a datastream.
3. After the negotiation is complete, the server starts sending data to the client.
4. The session ends by tearing down the control connection.

As shown here, the firewall needs to open a temporary conduit only when the client is on a less secure interface than the server. The port and IP addresses are extracted from the negotiation process. When TCP datastreams are used, after the initial connection to port 1755 is established, the client simply informs the server that it wants to use the same TCP connection for streaming, and the server starts sending data over the already established connection. There is no need for any extra processing by the firewall in this case (provided that access lists are set up correctly). NetShow application inspection is not configurable.

The VDO Live streaming protocol always uses two connections. The first is a TCP control connection established from the client to port 7000 on the server. The second is a UDP datastream from the server to the client. It always has a source port of 7001 and the destination port (the client-side port) is negotiated over the control connection during initial setup. The PIX monitors the VDO Live control connection and opens a temporary conduit for incoming traffic from port 7001 on the server to the negotiated port on the client. When the control connection is closed, the PIX closes the data connection as well. (There is no separate teardown message in this protocol, so this is the only way for the firewall to notice that communication has finished.) When NAT is involved, the PIX modifies the IP address and port number in the process of its negotiation correspondingly. Application inspection for VDO Live is not configurable and cannot be disabled.

SQL*Net

SQL*Net, which is used to query SQL databases, is another firewall-unfriendly protocol. There are three versions of SQL*Net: SQL*Net v1 (an old version used in Oracle 7), SQL*Net v2, and Net8/Net9 (newer versions of Oracle, such as 8i). Versions 1 and 2 are incompatible, whereas Net8/Net9 is just a small improvement on version 2. All these protocols have common behavior: When a client wants to connect to an Oracle server, it first establishes a connection to the dedicated Oracle port (port 1525 by default in SQL*Net version 1, port 1521 in

versions 2 and later) and then is redirected by this server to another instance of Oracle running on this machine or even another server. The client now has to establish a connection to the IP address and port it was told. In SQL*Net v2 and later, even after that the client can be redirected again.

The only case in which all communications happen only on one port without any redirection is when Oracle runs in Dedicated Server mode. This might need some extra configuration to function; refer to Oracle documentation if you are interested in this feature.

The problem with firewalls arises when the server is on a more secure interface than the client. Generally, the client will not be able to establish inbound connections to arbitrary ports and IP addresses. In order to process this correctly, the PIX needs to monitor the information exchange between the server and the client to notice which address/port number is negotiated and open a temporary conduit for inbound connections. The command for controlling application inspection of the SQL*Net protocol is:

```
[no] fixup protocol sqlnet [<port>[-<port>]]
```

The default port is 1521. In case of SQL*Net v1, the PIX scans all messages from the server to the client, checks the address and port negotiation, performs NAT on the embedded address if necessary, and forwards the resulting packets to the client. The inbound connections from the client are also de-NATted correctly and permitted by a temporary conduit.

SQL*Net version 2 communications are much more complicated than version 1, so the inspection process is also more complex. Messages used in this protocol can be of the following types: Data, Redirect, Connect, Accept, Refuse, Resend, and Marker. When the PIX firewall notices a Redirect packet with zero data length, it sets an internal flag for this connection to expect the relevant address/port information. This information should arrive in the next message, which must be only of Data or Redirect type. The relevant part of the message looks like the following:

```
(ADDRESS=(PROTOCOL=tcp)(DEV=6)(HOST=a.b.c.d)(PORT=p))
```

The PIX then needs to NAT this a.b.c.d:p pair inside the message and permit inbound connections on the corresponding IP address/port pair. If anything other than a Redirect or Data packet arrives after the initial null Redirect packet, the internal flag is reset.

H.323 and Related Applications

Voice over IP, or VoIP (including H.323 protocol set, SCCP, SIP, and others), is a real nightmare from both NAT and access control perspectives. VoIP applications use not one but many connections between the server and the client, initiate them in both directions, switch these connections, and embed address and port information in upper layers of communication that firewalls generally do not inspect. Here we look at various VoIP protocols and the degree to which they are supported by PIX application inspection features. All VoIP systems use two or three layers of application protocols, many protocols at the same time:

- **Signaling protocols (for system control and user information exchange)** SIP, MGCP, H.225 and RAS in H.323, SCCP.
- **Protocols for capabilities exchange** SDP, H.245.
- **Audio/media protocols (used for delivering speech and video)** RTP/RTCP.

H.323 can use up to two TCP connections and up to six UDP connections for a single call. Most of these are negotiated dynamically and do not use fixed ports. A basic H.323 call has the following sequence:

1. H.225 is used to initiate and terminate sessions between remote points (at least this connection has a fixed port number—TCP port 1720 by default). H.225 uses Registration, Admission and Status (RAS) protocol for certain authorization features (UDP ports 1718 and 1719).
2. During this process, a port for H.245 connection is negotiated.
3. The H.245 connection is used for negotiating port numbers for RTP/RTCP datastreams. (These ports can change during the call flow.)

H.323 version 2 provides a Fast Connect process, which, if used, eliminates the extra connection of H.245. H.245 messages, including RTP port negotiation, are transmitted over the same channel as initial H.225 connection.

NOTE

Support for H.323 version 2 was introduced in PIX firewall software version 5.3.

As with other application protocols, the PIX has the ability to inspect the negotiation process (for H.225, RAS, and H.245), remember the ports required for connection between parties, and perform NAT or PAT on the data portion of the packet. The two commands for controlling H.323 application inspection are:

```
[no] fixup protocol h323 h225 [<port>[-<port>]]
[no] fixup protocol h323 ras [<port>[-<port>]]
```

The first command is used for configuring ports that are monitored for H.225 messages (mainly for H.245 port negotiation), and the second is for ports on which RAS messages are intercepted. The default settings are:

```
fixup protocol h323 h225 1720
fixup protocol h323 ras 1718-1719
```

In PIX terms, "H.323 protocol inspection" means inspection of all protocols used in H.323 VoIP calls. The inspection of H.323 v2 was first implemented in PIX version 5.3. This was mainly the support of H.225 and H.245 inspection, including static or dynamic NAT on packet contents. RAS support was introduced in PIX firewall software version 6.2. This version also adds PAT support. Two major tasks performed by the PIX are:

- Monitoring and fixing of IP addresses and ports embedded in H.225, H.245, and RAS messages. These messages are encoded in PER format, so ASN.1 decoder is used internally.
- Opening the connections required for normal operations based on the preceding information.

Note that the first task is performed correctly even if messages are split into two or more packets—they are actually generally split in two packets, the first being a so-called TPKT header. When the PIX receives such a packet, it stores the information in an internal table, proxy ACKs this packet to the sender, and after receiving the next packet with IP address information, modifies necessary fields and sends out the modified message together with the new TPKT header. The PIX proxy feature does not support TCP options in the TPKT header.

UDP datastream connections are closed after the timeout period. This works in the same way as with general UDP packets, but you can use the following command to configure the timeout for datastreams separately from the general timeout:

```
timeout h323 <hh:mm:ss>
```

The default timeout is 5 minutes (this is the minimal setting), which is equivalent to:

```
PIX1(config)# timeout h323 0:5:0
```

NOTE

When RAS and gatekeepers are used, the initial setup is different. The client first sends an "Admission Request" (ARQ) UDP message, and the gatekeeper replies with an "Admission Confirmation" (ACF) message and provides the IP address and port number for a H.225 connection. There is no need to permit inbound traffic over port 1720 in this case; the PIX will open the necessary port based on inspection of the ACF message. Without gatekeepers, you need to enable incoming traffic to H.225 ports (1720 by default).

Besides hardware-based VoIP solutions, the H.323 set of protocols is also used by Intel Internet Phone, CU-SeeMe, CU-SeeMe Pro, MeetingPoint, and Microsoft NetMeeting.

CU-SeeMe is able to work in two different modes: H.323-compliant and native mode. Native mode is used when connecting to another CU-SeeMe client or CU-SeeMe conference server. The main difference here is that it uses a native control stream on UDP port 7648. The PIX performs inspection and NAT on this stream. CU-SeeMe support (other than support for H.323) is not configurable.

Skinny Client Control Protocol

Skinny Client Control Protocol (SSCP), as implied by its name, is a simplified protocol for use in VoIP networks. It is used by Cisco IP Phones. The main difference from full H.323 communications is that the whole session establishment is done not directly between clients but between a client and a Cisco Call Manager. After RTP ports are negotiated, datastreams are directly connected between clients. Thus, the PIX firewall needs to inspect SCCP signaling packets in order to note ports negotiated for RTP and possibly perform NAT on embedded addresses. The PIX firewall is able to recognize and inspect SCCP version 3.1.1. The relevant command is:

```
[no] fixup protocol skinny [<port>[-<port>]]
```

The default port number is 2000. NAT of SCCP messages is supported, whereas PAT is not. When the Cisco Call Manager is on a more secure interface than the phones, the IP phones can be configured to use TFTP to download the information used to connect to the Call Manager. (In most cases, the TFTP server runs on the same machine as the Call Manager.) The problem here is that the clients need to initiate an inbound TFTP connection (UDP port 69) to the server. To permit this connection, you need to either allow incoming traffic on port 69 to the TFTP server or create a static entry for this server without NAT, allowing external connections to its IP address. After clients download the configuration they need to contact the Call Manager, the rest of the traffic is controlled using SCCP application inspection.

Currently, the PIX firewall does not support fragmented SCCP messages because the application inspection process checks each received message for consistency and drops any messages with incorrect internal checksums. This usually happens when a single message is split into several TCP packets.

Session Initiation Protocol

Session Initiation Protocol (SIP), defined in RFC 2543, is another protocol used for session control in VoIP. It also uses SDP, mentioned previously, to describe each session being established. Each call is started with an *INVITE* message, which contains some of the session parameters, including IP addresses/ports for the next connections, which may use other ports. SDP messages then are used to establish RTP datastreams. The initial SIP session can use UDP or TCP as a channel. The default port for this connection is 5060. Application inspection of SIP over UDP is always on in the PIX and cannot be reconfigured. To change the default port for TCP SIP connections, use the following command:

```
[no] fixup protocol sip [<port>[-<port>]]
```

Application inspection for SIP includes monitoring of SIP and SDP messages, changing the IP addresses of endpoints embedded inside these messages (NAT and PAT), and opening temporary conduits for all negotiated control connections and datastreams based on the information obtained. The PIX maintains an internal database indexed by caller ID, sources, and destinations of each call. Included in this database are IP addresses and ports provided inside an SDP message. For example, a SIP message may look like the following (embedded address negotiation is in italics; these are the most important ones, although it includes much more IP information):

```
INVITE sip:23198@192.168.2.10:5060 SIP/2.0
Expires: 180
Content-Type: application/sdp
Via: SIP/2.0/UDP 192.168.2.10:5060;branch=1FV1xhfvxGJOK9rWcKdAKOA
Via:  SIP/2.0/UDP 10.0.1.134:5060
To:  <sip:23198@192.168.2.10>
From:  sip:15691@10.0.1.134
Call-ID:  c2943000-50405d-6af10a-382e3031@10.0.1.134
CSeq:  100 INVITE
Contact:  sip:15691@10.0.1.134:5060
Content-Length: 219
User-Agent:  Cisco IP Phone/ Rev. 1/ SIP enabled
Accept:  application/sdp
Record-Route: <sip:23198@192.168.2.10:5060;maddr=172.18.192.232>
```

The SDP message looks like the following:

```
v=0
o=CiscoSystemsSIP-IPPhone-UserAgent 17045 11864 IN IP4 10.0.1.134
s=SIP Call
c=IN IP4 10.0.1.134
t=0 0
m=audio 29118 RTP/AVP 0 101
a=rtpmap:0 pcmu/8000
a=rtpmap:101 telephone-event/8000
```

When the session setup starts, the SIP session is considered in a "transient" state until an RTP port has been negotiated for the datastream. If this does not happen within one minute, the session is discarded. After the RTP datastream ports are negotiated, the session is considered active and the SIP connection will remain established until the parties explicitly finish the call or an inactivity timeout expires. This timeout can be configured using the following command:

```
timeout sip <hh:mm:ss>
```

The default state of this timeout is 30 minutes, which is equivalent to the following setting:

```
PIX1(config)# timeout sip 0:30:0
```

RTP media connections are subject to a default timeout of 2 minutes, although this setting can be changed using this command:

```
timeout sip_media <hh:mm:ss>
```

You can view the status of SIP, RTP, and any of the connections subject to application inspection by PIX using the command:

```
show conn state
```

You can specify the type of connections you want to view (for example, *sip, h323, rpc*):

```
show conn state sip
```

NOTE

The PIX firewall supports PAT of SIP messages since version 6.2. NAT support has been available since version 5.3.

One issue that could require extra configuration with SIP occurs when a phone on a less secure interface tries to place on hold a phone on a more secure interface. This action is performed by the outside phone sending an extra *INVITE* message to the inside phone. If UDP is used as transport, the PIX will drop the incoming packet after the general UDP timeout has expired. This situation can be overcome either by configuring an access list on the outside interface that permits packets to port 5060/UDP on the inside gateway or by using the following command:

```
PIX1(config)# established udp 5060 permitto udp 5060 permitfrom udp 0
```

This command tells the PIX to allow inbound UDP packets to port 5060 on a client if it had outgoing communication from UDP port 5060.

Internet Locator Service and Lightweight Directory Access Protocol

Microsoft developed the Internet Locator Service (ILS) protocol for use in products such as NetMeeting, SiteServer, and Active Directory services. It is based on Lightweight Directory Access Protocol (LDAP) version 2. The main purpose of ILS application inspection is to let internal users communicate locally, even while

registered to outside LDAP servers. This is done by inspecting LDAP messages traversing the firewall and performing NAT when necessary. There is no PAT support, because only IP addresses are stored on the server. When attempting translation of an IP address, the PIX searches its internal XLATE table first, then DNAT tables. If neither contains the required address, it is left unchanged.

NOTE

If you use only *nat 0* (that is, you do not use NAT) and do not have DNAT communications, ILS fixup can be turned off safely. Turning it off will also improve the firewall's performance.

The command to configure application inspection for ILS is as follows:

```
[no] fixup protocol ils [<port>[-<port>]]
```

The default port is 389 (standard LDAP port). As with all other configurable inspection features, you can see the current configuration using the *show fixup* command.

ILS/LDAP communications occur on a client/server model over TCP, so there is no need for any temporary conduits to be opened by the PIX. During client/server communications, the PIX monitors for ADD requests and SEARCH responses, decoding them with BER decode functions; parses the message for IP addresses; translates them as necessary; encodes the message back, and sends the received packet to its destination.

Filtering Web Traffic

Although often the most attention is paid to the protection of internal servers or clients from external malicious attempts (the main purpose of ACLs), it is sometimes important to monitor and filter outbound connections made by users. One reason for content inspection is if you want to use your firewall to enforce security policies such as an acceptable use policy, which could specify that internal users may not use the company's Internet connection to browse certain categories of Web sites. There are many solutions for achieving this goal, but the most general one is URL filtering, in which the firewall hands each request for HTTP content to a filtering server, which can approve the request or deny access to it. The firewall then acts accordingly: If the request is approved, it is forwarded to

the outside server and the client receives the asked-for content; if not, either the request is silently dropped or the user is redirected to a page telling him or her that the request breaches company policy.

Another reason for filtering is to deal with "active content" such as ActiveX or Java applets. This could be important in order to protect internal users from malicious Web servers that embed these executable applets in their Web pages, because such executable content can contain viruses or Trojan horses. The most general solution is content filtering, which scans incoming applets for viruses and denies them when something wrong is found. Unfortunately, the PIX does not support this general solution, and the only thing you can do with it is to strip all active content from incoming Web pages.

Filtering URLs

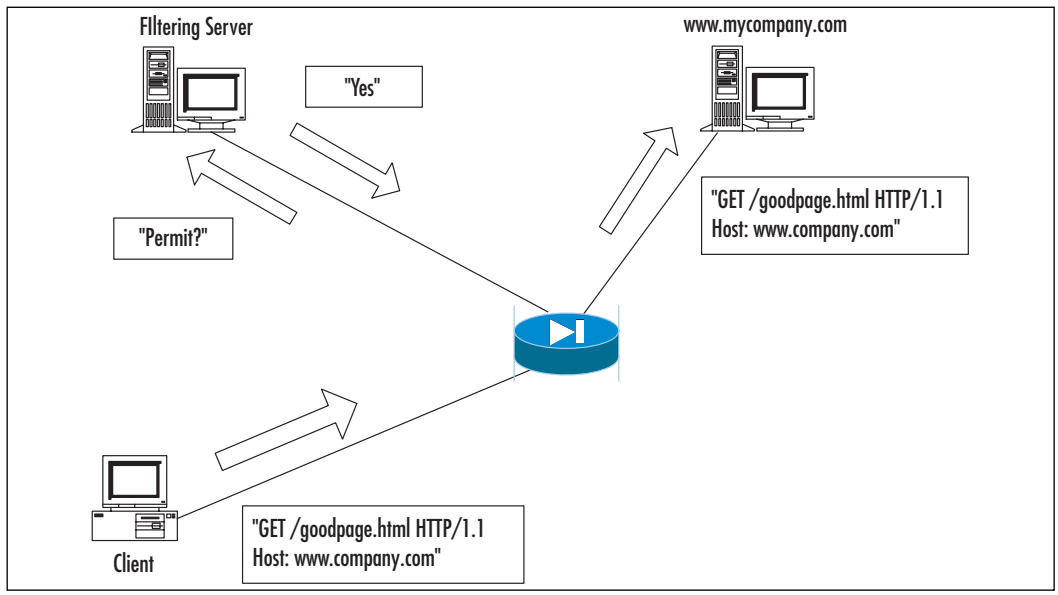
It is possible to use access lists to permit or deny access to specific Web sites, but if the list of sites grows long, this solution will affect firewall performance. In addition, access lists do not provide a flexible way of controlling access in this case; it is not possible, for example, to permit or deny access to specific pages on a Web site, only to the whole site identified by its IP address. Access lists will also not work for Web sites that are virtually hosted; in this case, there are many Web sites located on the same server and all of them have the same IP address, so it is only possible to deny or permit access to all of them at the same time.

As stated, one general solution moves most of the work to a dedicated URL filtering server, offloading the PIX's CPU and allowing for fine-tuning of Web access controls. The sequence of events is as follows:

1. A client establishes a TCP connection to a Web server.
2. The client sends an HTTP request for a page on this server.
3. The PIX intercepts this request and hands it over to the filtering server.
4. The filtering server decides if the client should be allowed access to the requested page.
5. If the decision is positive, the PIX forwards the request to the server and the client receives the requested content.
6. If the decision is negative, the client's request is dropped.

Figure 4.9 demonstrates this process.

Figure 4.9 Interaction Among a Client, a Web Server, PIX, and a Filtering Server



Websense and N2H2

The PIX can interact with two types of filtering servers: Websense (www.websense.com) and N2H2 (www.n2h2.com). Websense is supported in PIX version 5.3 and later, and N2H2 support was added in version 6.2. PIX URL filtering is applied only to HTTP requests; for example, it does not perform any inspections of FTP links. (Although a URL of type ftp://ftp.somedomain.com can be entered in a Web browser, it uses the FTP protocol, not HTTP.) The PIX also does not inspect HTTPS connections.

The steps to configure URL filtering are:

1. Specify the server to use for URL processing.
2. Tell the firewall the traffic to inspect—ports and IP addresses.
3. Optionally configure some server-specific parameters.
4. Configure filtering rules on the filtering server.

The command for specifying a filtering server for Websense is:

```
url-server (<if_name>) host <local_ip> [timeout <seconds>] [protocol
    <tcp> | <udp> [version 1|4]]
```

For example, the following code specifies that the PIX should use a server with IP address 10.0.0.1, which is located on the interface "inside," and connect to it using TCP Websense protocol version 4:

```
PIX1(config)# url-server (inside) host 10.0.0.1 protocol tcp version 4
```

Particularly, *if_name* is an interface on which the server is located, the default here is the inside interface. *local_ip* is the IP address of the filtering server. The PIX uses *timeout* (default is 5 seconds) to decide how long it has to wait for a reply from the server until it gives up and switches to the next configured server or takes a default action if there are no more servers available. It is possible to configure up to 16 servers, but they all must be of the same type; it is not possible to use both Websense and N2H2 filtering servers in the same configuration. The first server configured is a primary filtering server and is contacted first. Protocol type and version parameters specify the Websense protocol that should be used for communication with the server. It can be either TCP protocol version 1 (default) or 4 or UDP protocol version 4.

The N2H2 server is specified by the command:

```
url-server (if_name) vendor n2h2 host <local_ip> [timeout <seconds>]
    [port <port_number>] [protocol tcp | udp]
```

The meaning of parameters is the same. The parameter *vendor n2h2* states that the server is an N2H2 filtering server. It is possible to add the parameter *vendor websense* to the Websense server configuration, but it is assumed by default. N2H2 servers have only a communication protocol version available, so it is not specified. It is possible to configure the port to use for communication with the N2H2 server using the *port_number* parameter.

NOTE

If you switch the application type (that is, change from N2H2 server to Websense or vice versa), all configuration of URL filtering is lost and will need to be re-entered.

The next task is to configure the filtering policy itself. The relevant command is:

```
filter url <port>[-<port>] <local_ip> <local_mask> <foreign_ip>
    <foreign_mask> [allow] [proxy-block]
```

This command specifies port numbers on which HTTP connections should be inspected (with the default of port 80). *local_ip* and *local_mask* specify which local clients are subject to monitoring (that is, the requests by the machines from this network will be checked with URL filtering server). The *foreign_ip* and *foreign_mask* parameters specify that only requests to a specific set of servers be checked. The *allow* parameter defines that the PIX should permit traffic through if it is unable to contact the primary URL filtering server. Finally, the *proxy-block* parameter specifies that all requests from any clients to proxy servers will be denied. For example, the following command defines that all HTTP requests to port 80 will be inspected:

```
PIX1(config)# filter url http 0 0 0 0
```

The following command configures inspection of all HTTP requests to port 8080 from clients on network 10.100.1.0/24 to any server and allows the request to pass through in case a filtering server is unavailable:

```
PIX1(config)# filter url 8080 10.100.1.0 255.255.255.0 0 0 allow
```

Another variant of the *filter* command allows specifying that some traffic should be exempt from filtering. The format in this case is:

```
filter url except <local_ip> <local_mask> <foreign_ip> <foreign_mask>
```

When entered after the *filter* command, this command excludes specified traffic from the policy. For example, the following sequence of commands means that all HTTP traffic to port 8080 will be inspected, excluding traffic from network 10.100.1.0/24:

```
PIX1(config)# filter url 8080 0 0 0 0
PIX1(config)# filter url except 10.100.1.0 255.255.255.0 0 0 allow
```

Fine-Tuning and Monitoring the Filtering Process

The two commands we just looked at, *url-server* and *filter url,* constitute a basic configuration for URL filtering, but some extra parameters might need to be configured. One of these is required to deal with the problem of long URLs, which are common nowadays to store session and other information in the URL itself. A typical long URL could look like this:

```
http://www.somebettingcompany.com/?action=GoEv&class_id=1&type_id=2&ev_id=
    4288&class_name=%7CFootball%7C&type_name=%7CChampions+League%7C+%7C
        Qualifying+Matches%7C&ev_name=%7CGenk%7C+v+%7CSparta+Prague%7C
```

Until version 6.2, the PIX's maximum supported URL length was 1159 bytes (for Websense only; N2H2 was not supported at all). In version 6.2, the maximum URL length for Websense filtering is 6KB and 1159 bytes for N2H2. Version 6.2 introduced new options to the *filter* command to configure the firewall's behavior when the URL exceeds 1159 bytes with a Websense server. This syntax of this command is as follows:

```
filter url [longurl-truncate | longurl-deny] [cgi-truncate]
```

The *longurl-truncate* parameter specifies that when the URL length exceeds the maximum, only the IP address or hostname from the request, instead of the full URL, is sent to the filtering server. The *longurl-deny* parameter specifies that all long URL requests should be dropped. The *cgi-truncate* parameter specifies that only the CGI script name and its location (the part of the URL before the ? sign) should be passed as the URL to the Websense server. This skips the CGI parameter list, which can be quite long. Without this option enabled, the entire URL, including the parameter list, is passed.

NOTE

Even in PIX 6.2, the default URL size passed to a Websense filtering server for processing is 2KB. In order to increase this size, use the command *url-block url-size <size_in_kb>,* where *size_in_kb* can be from 2 to 6.

There are also commands for fine-tuning performance. The most important is the *url-cache* command:

```
url-cache {dst | src_dst} size <kbytes>
```

This command is used for tuning the process of caching replies from the filtering servers. By default, the PIX sends requests to the URL filtering server for a decision and to the Web server for content at the same time, and if the Web server replies faster than the filtering server, the Web server's reply is dropped. The Web server is then contacted again if the filtering server permits the connection. In order to prevent these double requests, you might want to store the filtering server replies locally instead of contacting the server every time. The *url-cache* command enables a cache of *kbytes* kilobytes for replies of filtering servers based either on destination (that is, Web server address) when the *dst* option is specified or on both source and destination when *src_dst* is specified. The first option is recommended when all users have the same access privileges (so there is no need

to identify clients), and the second is recommended when different users have different access privileges. The statistics of the caching process, including the hit ratio, can be viewed by executing the command:

```
show url-cache stat
```

For example, the following command enables a cache of 32KB for all outgoing HTTP requests:

```
PIX1(config)# url-cache dst size 32
```

The following are cache statistics:

```
PIX1# show url-cache stat
URL Filter Cache Stats
----------------------
Size : 32KB
Entries : 360
In Use : 200
Lookups : 2000
Hits : 1000
```

Another option for overcoming slow filtering server response is to cache Web server replies in advance and pass these replies to the client after the filtering server permits it. This feature is configured on the PIX using the following command:

```
url-block block <block_buffer_limit>
```

This command configures the size of the reply cache. The *block_buffer_limit* parameter can be any number between 1 and 128 and defines how many blocks of memory will be used. Usage statistics for this memory pool can be viewed by using the *show url-block block stat* command. For example:

```
pix(config)# show url-block block stat

URL Pending Packet Buffer Stats with max block            1
------------------------------------------------------------
Cumulative number of packets held: 0
Maximum number of packets held (per URL):                  0
Current number of packets held (global):                   0
Packets dropped due to exceeding url-block buffer limit:   0
Packet drop due to retransmission:                         0
```

The total amount of memory used for storing URLs and pending URLs (the ones for which no response from the filtering server has yet been received) is configured with the command:

```
url-block url-mempool <memory_pool_size>
```

The size of the allocated memory pool is defined by a number from 2 to 10240—the number in KB.

Other commands for viewing the configuration of URL filtering are:

```
show filter
show url-server
show url-server stats
```

Here is some example output from these commands:

```
PIX1# show url-server
url-server (outside) vendor n2h2 host 192.168.2.17 port 4005 timeout 5
protocol TCP
url-server (outside) vendor n2h2 host 192.168.2.10 port 4005 timeout 5
protocol TCP
PIX1# show filter
filter url http 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
PIX1# show url-server stats
URL Server Statistics:
----------------------
Vendor n2h2
URLs total/allowed/denied 2556/2000/556
URL Server Status:
------------------
192.168.2.17 UP
192.168.2.10 DOWN
```

The following monitoring commands can also be used for monitoring the performance of the URL filtering process:

```
show perfmon
show memory
show chunks
```

Active Code Filtering

As mentioned, active content in Web pages could be considered undesirable from a security point of view. Fortunately, there is a rather easy and effective way to prevent this content from reaching clients. In HTML, active content is denoted by two types of tags. The first is:

```
<object>
…
</object>
```

These tags are more common for ActiveX content, but they also can be used by Java applets. There are also Java-only tags:

```
<applet>
…
</applet>
```

When configured to look for active content, the PIX simply comments out both of these tags inside a TCP packet and the content between them, so they are simply skipped by the client's browser and embedded code is not run. The only problem with this approach is when the first tag is in one packet and the closing tag is in another packet, the PIX cannot perform this operation and the Web page is passed as is. For example, the HTML code inside an incoming packet might be as shown in Figure 4.10.

Figure 4.10 Packet Contents Before Being Changed by the PIX

```
<td width="185" height="68" valign="top">
  <applet codebase="/classes/" code="tscroll.class" align="absbottom"
       width="185" height="68">
    <param name="bgcolor" value="8,51,128">
    <param name="enddelay" value="4000">
    <param name="scrolldelay" value="25">
    <param name="scrolljump" value="5">
    <param name="speed" value="2">
    <param name="size" value="11">
    <param name="hlcolor" value="255,0,0">
    <param name="centertext" value="false">
  </applet>
</td>
```

After being transformed by PIX, it becomes the code in Figure 4.11.

Figure 4.11 Packet Contents After the Transformation

```
<td width="185" height="68" valign="top">
  <!– <applet codebase="/classes/" code="tscroll.class" align="absbottom"
      width="185" height="68">
    <param name="bgcolor" value="8,51,128">
    <param name="enddelay" value="4000">
    <param name="scrolldelay" value="25">
    <param name="scrolljump" value="5">
    <param name="speed" value="2">
    <param name="size" value="11">
    <param name="hlcolor" value="255,0,0">
    <param name="centertext" value="false">
  </applet> –>
</td>
```

Now the Web browser ignores everything between the *<td>* and *</td>* tags.

Filtering Java Applets

To configure filtering of Java applets, use the following command:

```
filter java <port>[-<port>] <local_ip> <mask> <foreign_ip> <mask>
```

Here is an example:

```
PIX1(config)# filter java 80 0 0 0 0
PIX1(config)# filter java 80 192.168.2.17 255.255.255.255 0 0
```

The first command configures the PIX to drop all Java applets from incoming Web pages; the second prohibits only one host 192.168.2.17 to download Java applets. The *port* parameter, as usual, specifies the TCP port on which to perform the inspection.

Filtering ActiveX Objects

Java has a more or less robust security model for its active code (there has been only one big security issue with it, and that was due to the poor implementation of this model in some versions of Netscape), but ActiveX objects have almost unrestricted access to the client's machine.

The command to configure filtering of ActiveX code (and all active content that is embedded in "object" tags) is very similar to Java filtering:

```
filter activex <port>[-<port>] <local_ip> <mask> <foreign_ip> <mask>
```

Here is an example:

```
PIX1(config)# filter activex 80 0 0 0 0
```

This command configures the PIX to comment out all pairs of "object" tags from all incoming Web pages, disabling ActiveX and some Java applets.

Configuring Intrusion Detection

One of important features of the PIX firewall is its intrusion detection capability. Cisco has a dedicated IDS product called Cisco Secure IDS (former NetRanger appliance), but a limited part of its functionality is implemented in both Cisco IOS and Cisco PIX. Because the PIX is basically an OSI Layers 3 and 4 filtering device, it supports detection of only simpler attacks that happen on these layers of network communication and can be detected by inspecting a single packet in the traffic. The IDS signatures (that is, descriptions of attacks) that the PIX supports are a subset of the Cisco Secure IDS signature set and are embedded in PIX software. In order to upgrade this set of signatures, you need to upgrade the whole PIX firmware using a general upgrade procedure. Doing so does not pose a big problem, though, because these signatures describe very general and simple attacks, which are not invented often. Intrusion detection can be configured on each interface in inbound and outbound directions. When the PIX detects each signature, the device produces an alert (the alert can be of two types, "information" or "attack," depending on the severity of the attack) and sends it via syslog to the configured destination.

Supported Signatures

Unfortunately, Cisco's own documentation is not quite clear about signatures supported in each specific version. The best way to check what your PIX can do in the area of intrusion detection is to browse a list of syslog messages produced by the specific version (for example, see the *Cisco PIX Firewall System Log Messages* guide). For version 6.2, syslog messages numbered from 400 000 to 400 050 are reserved for IDS messages. Their format is shown here:

```
%PIX-4-4000<nn>: : <sig_num> <sig_msg> from <IP_addr> to <IP_addr> on
    interface <int_name>
```

This syslog message means that PIX has detected an attack with number *sig_num* and name *sig_msg*. The two IP addresses show the origin and the destination of this attack. Finally, the interface on which the attack was detected is mentioned. For example:

```
%PIX-4-400013 IDS:2003 ICMP redirect from 1.2.3.4 to 10.2.3.1 on
    interface dmz
```

Table 4.2 lists all signatures detected by PIX, with short descriptions.

Table 4.2 PIX IDS Signatures

Message Number	Signature ID	Signature Title	Signature Type
400000	1000	IP options-Bad Option List	Informational
400001	1001	IP options-Record Packet Route	Informational
400002	1002	IP options-Timestamp	Informational
400003	1003	IP options-Security	Informational
400004	1004	IP options-Loose Source Route	Informational
400005	1005	IP options-SATNET ID	Informational
400006	1006	IP options-Strict Source Route	Informational
400007	1100	IP Fragment Attack	Attack
400008	1102	IP Impossible Packet	Attack
400009	1103	IP Fragments Overlap	Attack
400010	2000	ICMP Echo Reply	Informational
400011	2001	ICMP Host Unreachable	Informational
400012	2002	ICMP Source Quench	Informational
400013	2003	ICMP Redirect	Informational
400014	2004	ICMP Echo Request	Informational
400015	2005	ICMP Time Exceeded for a Datagram	Informational
400016	2006	ICMP Parameter Problem on Datagram	Informational
400017	2007	ICMP Timestamp Request	Informational
400018	2008	ICMP Timestamp Reply	Informational
400019	2009	ICMP Information Request	Informational
400020	2010	ICMP Information Reply	Informational

Continued

Table 4.2 Continued

Message Number	Signature ID	Signature Title	Signature Type
400021	2011	ICMP Address Mask Request	Informational
400022	2012	ICMP Address Mask Reply	Informational
400023	2150	Fragmented ICMP Traffic	Attack
400024	2151	Large ICMP Traffic	Attack
400025	2154	Ping of Death Attack	Attack
400026	3040	TCP NULL flags	Attack
400027	3041	TCP SYN+FIN flags	Attack
400028	3042	TCP FIN only flags	Attack
400029	3153	FTP Improper Address Specified	Informational
400030	3154	FTP Improper Port Specified	Informational
400031	4050	UDP Bomb attack	Attack
400032	4051	UDP Snork attack	Attack
400033	4052	UDP Chargen DoS attack	Attack
400034	6050	DNS HINFO Request	Attack
400035	6051	DNS Zone Transfer	Attack
400036	6052	DNS Zone Transfer from High Port	Attack
400037	6053	DNS Request for All Records	Attack
400038	6100	RPC Port Registration	Informational
400039	6101	RPC Port Unregistration	Informational
400040	6102	RPC Dump	Informational
400041	6103	Proxied RPC Request	Attack
400042	6150	ypserv (YP server daemon) Portmap Request	Informational
400043	6151	ypbind (YP bind daemon) Portmap Request	Informational
400044	6152	yppasswdd (YP password daemon) Portmap Request	Informational
400045	6153	ypupdated (YP update daemon) Portmap Request	Informational
400046	6154	ypxfrd (YP transfer daemon) Portmap Request	Informational

Continued

Table 4.2 Continued

Message Number	Signature ID	Signature Title	Signature Type
400047	6155	mountd (mount daemon) Portmap Request	Informational
400048	6175	rexd (remote execution daemon) Portmap Request	Informational
400049	6180	rexd (remote execution daemon) Attempt	Informational
400050	6190	statd Buffer Overflow	Attack

The signature IDs listed in the table correspond to signature numbers on the Cisco Secure IDS appliance. See www.cisco.com/univercd/cc/td/doc/product/iaabu/csids/csids1/csidsug/sigs.htm (*Cisco Secure Intrusion Detection System Version 2.2.1 User Guide*) for a complete reference. All signatures are divided into two classes: informational and attack. The division is rather deliberate and cannot be changed, but it makes sense most of the time. For example, all DoS attacks are listed as attacks, and all information requests only have informational status. You might feel that if somebody tries to obtain information on RPC services on one of your hosts, this constitutes an attack, but it is still listed as informational by Cisco. Generalizing a little, it is possible to suggest the following reasoning on attack classification (from top to bottom in the table):

- Packets with IP options will not do any harm because they are always dropped by the PIX, so if these packets are detected, send only an informational message.
- Fragmented packets can pass through the firewall and are generally difficult to inspect, so they constitute an attack attempt.
- Legitimate ICMP traffic, although unwanted and maybe revealing some information about your network (for example, ICMP Information Request), is not classified as an attack.
- Fragmented ICMP, Ping of Death, and so on are considered attacks.
- Impossible TCP flag combinations are considered attacks because they are sometimes used for stealth scanning of networks.
- All floods/DoS attempts (including the UDP Snork attack) are classified as attacks.

- DNS transfers are classified as attacks; they reveal too much about the network.
- General RPC requests and all information requests for various RPC services are not considered that harmful and are classified as informational.
- Some specific one-packet attacks on RPC services are recognized separately.

Configuring Auditing

Auditing is configured using the *ip audit* command. Auditing can be turned on or off, different auditing policies can be created, the policies can be applied to specific interfaces, and specific signatures can be turned on or off. The easiest configuration requires you to assign a name for the auditing policy, specify actions (one for informational signatures and one for attack signatures) to be taken, and apply the policy to an interface. The actions that can be taken are:

- **Alarm** When PIX detects a signature in the packet, it reports with the message described previously to all configured syslog servers.
- **Drop** When this action is configured, PIX drops the offending packet.
- **Reset** This action means that PIX should drop the packet and close the connection if this packet was a part of an open connection.

The default action is alarm. Policy configuration usually takes no more than two commands:

```
ip audit name <audit_name> info action [drop | alarm | reset ]
ip audit name <audit_name> attack action [drop | alarm | reset ]
```

For example, the following commands create a policy with the name *myaudit* and specify that when an informational signature is matched, the PIX should send an alarm to syslog, and when an attack signature is matched, the PIX should drop the packet:

```
PIX1(config)# ip audit name myaudit info action alarm
PIX1(config)# ip audit name myaudit attack action drop
```

It is possible to omit the *action* in the configuration. In this case, the default action is applied. Default actions are configured via these commands:

```
ip audit info action [drop | alarm | reset ]
ip audit attack action [drop | alarm | reset ]
```

If not changed, the default action is *alarm*. Note that if you issue only the following command but not the corresponding *attack* command, no attack signatures will be matched:

```
PIX1(config)# ip audit name myaudit info action alarm
```

On the other hand, if you configure the policy in the following manner, omitting the action for informational signatures, both informational and attack signatures will be matched, and the default action (alarm) will be applied when a packet is matched with an informational signature:

```
PIX1(config)# ip audit name myaudit info
PIX1(config)# ip audit name myaudit attack action drop
```

After creating a policy, you need to apply it to an interface in order to activate IDS on the interface. For example:

```
PIX1(config)# ip audit interface outside myaudit
```

This means that all signatures and actions configured should be matched on the outside interface. The general form of this command is:

```
ip audit interface <if_name> <audit_name>
```

- *if_name* is the name of an interface where the IDS has to check for packets.
- *audit_name* is a name of the policy that describes which actions to take.

As an example, let's configure a simple IDS on the outside interface, which will send an alarm when an informational signature is matched and drop the connection when an attack is noticed:

```
PIX1(config)# ip audit name myaudit info alarm
PIX1(config)# ip audit name myaudit attack action drop
PIX1(config)# ip audit interface outside myaudit
```

Each command has its *no* equivalent, which removes the command from the configuration. For example:

```
PIX1(config)# no ip audit interface outside myaudit
PIX1(config)# no ip audit name myaudit info
```

Another command allows easy clearing of all IDS configuration related to an interface, policy, or default action:

```
clear ip audit [name | signature| interface | audit | info | attack ]
```

The following set of commands displays the corresponding configuration of IDS related to the interface, audit, or default action. This code simply shows the commands you entered when configuring these parameters:

```
show ip audit interface <if_name>
show ip audit info
show ip audit attack
show ip audit name <audit_name>
```

Disabling Signatures

Imagine the following situation: You are interested in being alarmed on the informational signature 6102, "RPC Dump." This means that you have to include all informational signatures in your policy with a command such as:

```
PIX1(config)# ip audit name myaudit info action alarm
```

Here comes the problem: Many other signatures are listed as informational, and some of them are very "noisy"—generating lots of alarms—for example, number 2000, "ICMP echo reply," which is simply a response to a ping. Chances are, you will be flooded with alarms on this latter signature and will not notice the former one, which is the one in which you are actually interested. One way to get around this issue is to disable the noisy signatures with the following command, which disables the detection of the signature with number *sig_number*:

```
ip audit signature <sig_number> disable
```

In our case, to disable the "ICMP echo reply" signature, use the following command:

```
PIX1(config)# ip audit signature 2000 disable
```

After this command is entered, signature number 2000 ("ICMP echo reply") will not be detected by the PIX at all. Note that disabling a signature means disabling it globally, not for a specific interface or audit.

It is possible to see the list of all disabled signatures with the command:

```
PIX1(config)# show ip audit signature
```

You can enable a disabled signature with a *no* command in Configuration mode:

```
no ip audit signature <sig_number> disable
```

Configuring Shunning

Shunning is a term used in the IDS context to describe blocking traffic from an attacking host; it is configured on the PIX using the following command:

```
shun <src_ip> [<dst_ip> <sport> <dport> [<protocol>]]
```

This technique temporarily blocks all traffic from the specified source IP address. To block all traffic, the source IP address of 10.0.1.1, use the following command:

```
PIX1(config)# shun 10.0.1.1
```

You can also deny specific traffic from the source IP by specifying a source port, destination IP address, and destination port number. After the *shun* command is entered, the PIX deletes all matching connections from its internal connection table and drops all further packets that match the command's parameters. The action of this command takes priority over access list entries and even security levels on interfaces; all specified traffic is blocked, whether the offending host is on the inside or outside of the interface. In order to remove this blocking action, use the corresponding *no* command. For example:

```
PIX1(config)# no shun 10.0.1.1
```

This command is dynamic and is not displayed or stored in the configuration. If you want to view active shuns, use the *show shun* command. The *clear shun* command deletes all shun entries.

DHCP Functionality

As more Cisco devices are used in SOHO environments, it becomes more important that they support features such as Dynamic Host Configuration Protocol (DHCP). Hosts use DHCP to dynamically obtain their Internet configuration instead of being configured with a static IP address and other parameters. The operation is very simple: Upon connection, a client sends a UDP broadcast, and if receives a specific reply, it configures itself correspondingly. Of course, this works only on the directly connected LAN segment or on the segments that are connected through bridges or routers, which forward broadcasts. This method can be used, for example, to simplify workstation management; all reconfigurations will be carried on only on the DHCP server itself, which will provide the new configuration to the workstations.

The Cisco PIX firewall can act both as a DHCP server and a client. In the first case, it will probably be a gateway for a small network of workstations and provide them all the information they need in order to connect to the Internet. In its client role, it may be a gateway for a network connected through a dialup line, acquiring its outside interface address from the ISP's DHCP server.

Although DHCP functionality on the PIX firewall is available on all models of hardware, it was specifically designed for PIX 501, 506, and 506E, which are used primarily in SOHO environments. This is why the DHCP features the PIX firewall offers have some limitations. For example, the DHCP server can only support a maximum of 256 clients (or even fewer, depending on the firewall model, version, and license). There is also no BOOTP support and no failover support; the current state of DHCP server or client is not replicated over failover link.

DHCP Clients

When configured as a DHCP client, the PIX firewall can obtain the configuration of its outside interface from a designated DHCP server—for example, a server located at an ISP. This configuration includes the IP address, the subnet mask, and optionally, the default route.

NOTE

The DHCP client feature can only be configured on the "outside" interface of the PIX firewall.

This address can be used, for example, as a PAT address for all outgoing communications. This is configured in the following way (assuming that the DHCP client is already configured):

```
nat (inside) 1 0 0
global (outside) 1 interface
```

This configuration will work with any IP address assigned to the outside interface by DHCP.

The configuration of the DHCP client is rather simple, and all you need to use is the following command:

```
ip address outside dhcp [setroute] [retry <retry_cnt>]
```

You do this instead of specifying a fixed IP address for an outside interface. The optional *setroute* keyword forces the PIX firewall to pick up not only the IP address and the subnet mask but the default route as well. Do not configure a static default route on the firewall if you use the *setroute* option. The *retry* option tells the PIX firewall to try to contact a DHCP server a specified number of times before giving up. If this keyword is not specified, no retries are attempted. If this keyword is specified but no retry count is given, the default number of retries is four. For example, the following command configures a DHCP client on the outside interface to obtain an IP address, subnet mask, and default route from the DHCP server, and only one attempt will be made:

```
PIX1(config)# ip address outside dhcp setroute
```

The following command configures the DHCP client to obtain an IP address and subnet mask only and tries at least five times before giving up if no DHCP servers are available:

```
PIX1(config)# ip address outside dhcp retry 5
```

There are no special commands for renewing and releasing DHCP lease; simply issue the same command again and the lease will be renewed.

The address obtained can be viewed using:

```
PIX1# show ip address outside dhcp
```

This produces output similar to the following:

```
Temp IP Addr:123.1.2.3 for peer on interface:outside
Temp sub net mask:255.255.255.0
DHCP Lease server:123.1.2.31, state:3 Bound
DHCP Transaction id:0x4567
Lease:259200 secs, Renewal:129600 secs, Rebind:226800 secs
Temp default-gateway addr:123.1.2.1
Next timer fires after:100432 secs
Retry count:0, Client-ID:cisco-0000.0000.0000-outside
```

This output means that PIX has obtained an IP address of 123.1.2.3 and a subnet mask of 255.255.255.0 from the DHCP server 123.1.2.31. This DHCP lease is granted for 259200 seconds with renewal time of 129600 seconds. Time left until the next renewal is 100432 seconds, and there were no retries in contacting the server.

In case there are any issues with the DHCP client, you can troubleshoot using *debug* commands:

```
debug dhcpc packet
debug dhcpc detail
debug dhcpc error
```

These are self-explanatory. *debug dhcpc packet* displays all DHCP traffic between the PIX client and a remote server, the *detail* option shows details of negotiation, and the *error* option displays all errors in this communication.

DHCP Servers

The server part of PIX DHCP support is more complicated. Let's look at the server's abilities and limitations. The most important issue is the number of DHCP clients the server can support and the specific protocol options supported. The number of clients supported on the various versions of PIX firewalls is shown in Table 4.3.

Table 4.3 Number of Clients Supported by the PIX DHCP Server

PIX Firewall Version	PIX Firewall Platform	Client Addresses (Active Hosts)
Version 5.2 and before	All platforms	10
Version 5.3 to version 6.0	PIX 506/506E	32
	All other platforms	256
Version 6.1 and after	PIX 501 with 10-user license	32
	PIX 501 with 50-user license	128
	All other platforms	256

Note that the numbers quoted in Table 4.3 are for active hosts. A host is "active" if it has passed any traffic through the PIX, established a connection through the firewall, established a NAT or PAT translation entry, or authenticated itself to the firewall during the last 30 seconds.

NOTE

The DHCP server can be configured only on the inside interface of the PIX firewall and supports only clients on a network directly connected to this interface.

A minimal configuration of the DHCP server requires only two commands: one for specifying a range of IP addresses that can be provided to clients and another one for actually turning the feature on. For example:

```
PIX1(config)# dhcpd address 192.168.2.1-192.168.2.127 inside
PIX1(config)# dhcpd enable inside
```

The only parameter that can be changed here is the address pool. Although currently the interface is always *inside*, it is possible that future releases of the PIX will have the ability to run a DHCP server on other interfaces. However, at the time of this writing (version 6.2), it does not. It is possible to configure only one pool. Now when a client sends a DHCP request, the PIX provides it with the next IP address available in the pool of 192.168.2.1-192.168.2.127, the same subnet mask that is set for the inside interface of the firewall, and a default route pointing to PIX itself.

Some other configuration parameters are concerned with so-called "DHCP options"—optional information that can be provided to the client by its request. RFC 2132, "DHCP Options and BOOTP Vendor Extensions," describes about 100 of these options and provides a mechanism for vendors to specify their own options. Very few of these options are really needed, especially in a SOHO environment, so the PIX supports only a few of them; nevertheless, this does not make it unable to operate as a full-strength server. The options that can be configured are the default domain name, the DNS server, the WINS server, and two TFTP-related options (number 66 and 150).

The domain name provided to a client is configured with the following command:

```
dhcpd domain <domain_name>
```

For example:

```
PIX1(config)# dhcpd domain syngress.com
```

The DNS servers that a client should use are configured with the command:

```
dhcpd dns <dns1> [<dns2>]
```

Up to two DNS servers can be configured, using IP addresses:

```
PIX1(config)# dhcpd dns 1.2.3.4 1.2.4.10
```

WINS servers are configured using the following command, with the same restrictions as DNS servers—up to two servers, configured using IP addresses:

```
dhcpd wins <wins1> [<wins2>]
```

Options 66 and 150 are used mostly by Cisco IP Phones and are considered later in this chapter. Other DHCP-related commands allow specifying some internal parameters for the server. It is possible to change the default lease time (the amount of time for which an IP address is provided to the client):

```
dhcpd lease <lease_time>
```

This command specifies the time in seconds. The default value is 3600, and possible values are from 300 seconds to 2,147,483,647 seconds. The following command sets a maximum ping timeout in milliseconds (1/1000th of a second):

```
dhcpd ping_timeout <ping_time>
```

The PIX uses *ping* to ensure that another host on the network does not already have the IP address it is about to grant. If no host with this IP replies during this timeout, the IP is considered free. The *ping* timeout specifies how long the PIX will wait for a *ping* response to ensure that a host with the same IP address does not already exist on the network.

Finally, the following command allows the DHCP server to automatically obtain DNS, WINS, and domain parameters from a DHCP client configured on the outside interface:

```
PIX1(config)# dhcpd auto_config outside
```

An example of a SOHO configuration follows. It includes a DHCP client on the outside interface and a DHCP server on the inside interface, and it passes parameters from the client to the server:

```
ip address outside dhcp setroute
PIX1(config)# ip address inside 192.168.2.1 255.255.255.0
PIX1(config)# dhcpd address 192.168.2.201-192.168.2.210
PIX1(config)# dhcpd lease 3000
PIX1(config)# dhcpd auth_config outside
PIX1(config)# dhcpd enable
PIX1(config)# nat (inside) 1 0 0
PIX1(config)# global (outside) 1 interface
```

Without auto configuration, the example may look like this:

```
PIX1(config)# ip address outside dhcp setroute
PIX1(config)# ip address inside 192.168.2.1 255.255.255.0
PIX1(config)# dhcpd address 192.168.2.201-192.168.2.210
PIX1(config)# dhcpd lease 3000
PIX1(config)# dhcpd dns 1.2.3.4 1.2.3.31
PIX1(config)# dhcpd wins 192.168.2.20
```

```
PIX1(config)# dhcpd domain example.com
PIX1(config)# dhcpd enable
PIX1(config)# nat (inside) 1 0 0
PIX1(config)# global (outside) 1 interface
```

Commands are available for checking the state of the server. For example:

```
PIX1(config)# show dhcpd
dhcpd address 192.168.2.201-192.168.2.210 inside
dhcpd lease 3000
dhcpd ping_timeout 750
dhcpd dns 1.2.3.4 1.2.3.31
dhcpd enable inside
```

Other commands show the current state of IP bindings (which client has been assigned which IP address) and general server statistics:

```
PIX1(config)# show dhcpd binding
IP Address Hardware Address Lease Expiration Type
192.168.2.210 0100.a0c9.777e 84985 seconds automatic
```

Here, a client with MAC address 0100.a0c9.777e has obtained IP address 192.168.2.210, and this lease will expire in 84985 seconds:

```
PIX1(config)# show dhcpd statistics
Address Pools 1
Automatic Bindings 1
Expired Bindings 1
Malformed messages 0
Message Received
BOOTREQUEST 0
DHCPDISCOVER 1
DHCPREQUEST 2
DHCPDECLINE 0
DHCPRELEASE 0
DHCPINFORM 0
Message Sent
BOOTREPLY 0
DHCPOFFER 1
DHCPACK 1
DHCPNAK 1
```

These statistics show the number of IP address pools configured, the number of active leases (bindings), expired bindings, messages received with errors, and a detailed breakdown on message type for correctly received and sent messages.

Cisco IP Phone-Related Options

As described in the "Skinny Client Control Protocol" section, Cisco IP Phones use a TFTP server for obtaining most of their configuration. This address can be configured statically, but it is also possible to use special DHCP options in order to provide phones with the location of the TFTP server. Clients can send to DHCP servers messages with options of two types: number 66, which causes the server to send a name of one TFTP server, and option 150, which results in a list of IP addresses of one or two TFTP servers. These options are supported starting from version 6.2 of PIX software and are configured with the following commands:

```
dhcpd option 66 ascii <server_name>
dhcpd option 150 ip <server1_ip> [<server2_ip>]
```

For example:

```
PIX1(config)# dhcpd option 66 ascii tftp.example.com
PIX1(config)# dhcpd option 150 ip 1.2.3.4 2.3.4.5
```

Because the server runs only on the inside interface, IP Phones should be placed on the network directly connected to this interface.

Other Advanced Features

The Cisco PIX firewall has many other security features. Some of these features can be used in order to protect the network against various DoS attacks. Some of them are related to the processing of routing information—both unicast and multicast.

Fragmentation Guard

Fragmented packets are a challenge to firewalls. For example, nothing in the current Internet standards prevents a person from sending IP packets so fragmented that IP addresses of source and destination and TCP port information are located in different fragments or even in overlapping fragments. The firewall cannot decide on what to do with the packet until it sees the entire TCP/IP header. Some firewalls simply pass the fragments without trying to reassemble the

original packets, whereas others try to perform this reassembly. Reassembly can be a dangerous process—for example, it is very easy to send fragments that will cause the reassembled packet to be of illegal size, possibly crashing internal buffers of the IP stack implementation.

The PIX always performs reassembly of fragmented packets before they are checked against access lists and can impose some restrictions on the fragmented traffic that passes through it. The FragGuard feature, when turned on, ensures that:

- Each noninitial IP fragment is associated with an already seen initial fragment (teardrop attack prevention).
- The rate of IP fragments is limited to 100 fragments per second to each internal host.

This feature theoretically breaks some rules of processing fragmented packets, but the current state of the Internet is such that heavy fragmentation usually does not occur naturally and almost always is the result of a malicious hacker trying to circumvent firewall rules or flood an Internet host. Therefore, in general, it is much better to have this feature on, unless you are connected via some strange link, which does have a lot of fragmentation—but again, in this case there might be something wrong with the link itself.

This feature is disabled by default and can be turned on or off on all interfaces simultaneously only. The command for enabling it is:

```
sysopt security fragguard
```

The corresponding *no* command turns the feature off. The status of various settings, including FragGuard, can be checked with the *show sysopt* command.

NOTE

The most important side effect of FragGuard is that you could loose the communication with hosts running some versions of Linux if they do fragment IP packets. These versions do not always send the initial fragment first, so the PIX firewall will discard the received sequence of fragments. Although this rarely occurs, you should still watched out for it.

FragGuard settings can be too restrictive at times. It is possible to manually tune the process of virtual reassembly with the *fragment* set of commands. Their syntax is as follows:

```
fragment size <database-limit> [<interface>]
fragment chain <chain-limit> [<interface>]
fragment timeout <seconds> [<interface>]
clear fragment
```

The first command sets the maximum number of blocks that can be used for fragment reassembly. If an interface is not specified, the setting is global; otherwise, this setting is for the specific interface. The default number of blocks is 200 and should never be greater than the total number of available blocks of 1550 bytes' size. In general, a bigger database makes PIX more vulnerable to a DOS attack by flooding it with fragments and exhausting its memory.

The second command sets the maximum allowed number of fragments into which one IP packet is split. The default setting is 24 fragments; the maximum is 8200. Further fragments will be discarded and the packet will not be reassembled. The timeout setting specifies the time frame in which all fragments of one IP packet should be received. The default timeout is 5 seconds and can be up to 30 seconds.

The last command, *clear fragment*, resets all three settings to their default values. The state of fragments database can be displayed with the *show fragment* command:

```
pix(config)# show fragment outside
Interface:outside
Size:200, Chain:24, Timeout:5
Queue:150, Assemble:300, Fail:0, Overflow:0
```

This output shows that the database has default settings: the size of 200 blocks, 24 fragments in a chain, 5-second timeout. There are 150 packets waiting to be reassembled, 300 were already successfully reassembled, and there were no failures or database overflows.

AAA Floodguard

Another flood-related problem is that somebody can abuse the PIX AAA authentication mechanism simply by making a large number of login attempts without providing any login information, leaving the connections open. The PIX firewall will then wait until a timeout expires. By making enough attempts, it is possible to exhaust AAA resources so that no further login attempts will be answered—a DoS on login resources. In order to prevent this situation, the PIX firewall has an internal mechanism for reclaiming AAA resources. It is called Floodguard and is enabled by default. When enabled, Floodguard causes the PIX firewall to monitor

resource usage and send a syslog message when these resources are exhausted. When in need of additional resources, the PIX firewall will reclaim the ones that are not in active state. This is done in the following order (by priority):

1. Resources that are in the Timewait state are reclaimed.
2. Resources in the Finwait state are reclaimed.
3. Embryonic resources are reclaimed.
4. Idle resources are reclaimed.

Commands (Configuration mode) related to this feature are quite simple:

```
floodguard enable
floodguard disable
show floodguard
```

These commands are self-explanatory.

SYN Floodguard

Another well-known DoS attack is SYN flooding, which occurs when an attacker sends large numbers of initial SYN packets to the host and neither closes nor confirms these half-open connections. This causes some TCP/IP implementations to use a great deal of resources while waiting for connection confirmation, preventing them from accepting any new connections before the backlog of these half-open connections is cleared. The easiest way to prevent this from happening is to control the rate at which new connections are opened or the number of connections that are half-open (other names for this are *SYN Received* or *embryonic*) at any given time. The latter can be performed by specifying a limit on the number of embryonic connections in the *static* and *nat* configuration commands. For example:

```
PIX1(config)# static (dmz, outside) 123.4.5.6 10.1.1.0 netmask
    255.255.255.255 100 50
```

This creates a static NAT entry for the DMZ server 10.1.1.0 with an external IP address of 123.4.5.6. The number 100 means that only 100 connections to this server from outside can be in an open state at any given time, and the number 50 is the number of half-open or embryonic connections to this server that can exist at any given time. The *nat* command is similar: Two numbers at the end specify

the number of open and embryonic connections that can exist at any given time to each translated host:

```
nat (inside) 1 10.0.0.0 255.0.0.0 100 50
```

When any of these numbers is zero, the number of connections is not limited. The actual behavior of PIX when the number of embryonic connections is reached for a host is different in versions 5.2 and later (since 5.3); see the sidebar for details.

Figure 4.12 illustrates how the TCP Intercept feature works.

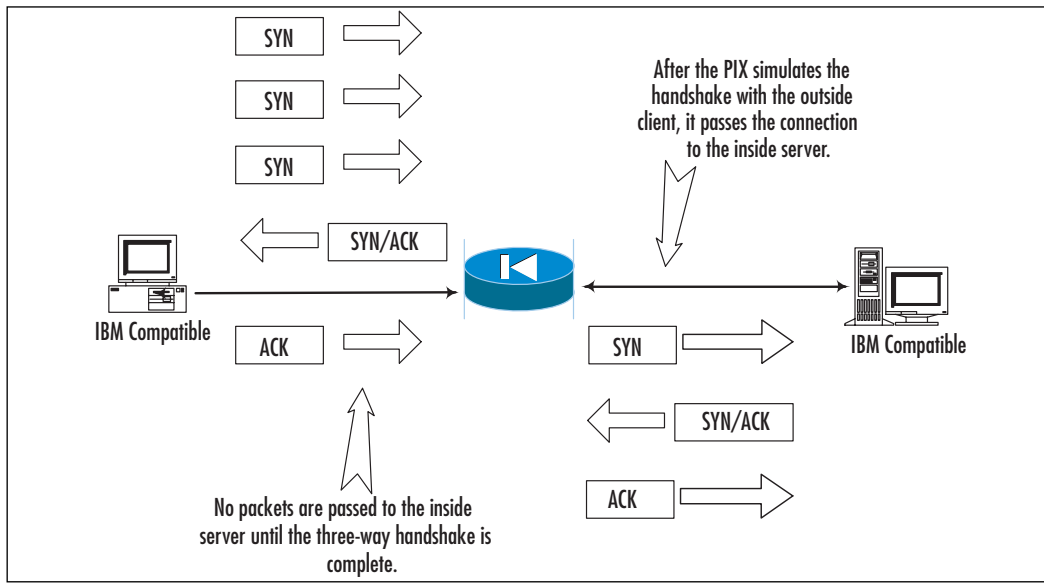
Designing & Planning…

The TCP Intercept Feature in PIX Version 5.3 and Later

The implementation of SYN Floodguard in versions before 5.3 was not quite good. When the maximum number of embryonic connections for a host was reached, the PIX firewall simply discarded any further SYN packets directed to the affected host. Thus, while protecting the host against overloading, the PIX firewall prevented any traffic from passing to or from the host in the case of a SYN flood. Similarly, when the maximum number of embryonic connections was not specified, the PIX did not restrict the number of half-open connections, which could lead to a successful SYN flood attack against the host.

Version 5.3 implements a new feature called *TCP Intercept*. Since version 5.3, the PIX firewall behaves differently when the number of embryonic connections for a host is reached. If this happens, until the number of embryonic connections falls below threshold, each new SYN packet to the affected host is intercepted instead of being discarded. Then PIX itself replies to the sender instead of the destination server with SYN/ACK. If the client finally replies with a legitimate ACK, the PIX firewall sends the original SYN to its destination (the server), performs a correct three-way handshake between the PIX and the server, and the connection is resumed between a client and a server.

Figure 4.12 TCP Intercept in PIX Versions 5.3 and Later



Reverse-Path Forwarding

The concept of reverse-path forwarding (RPF) is rarely understood well, although it is rather simple. The basic idea is to have an extensive routing table and, for each packet arrived, check its *source* address against this table. This is why it is called "reverse" lookup. When a route to this source is found (that is, when there is a reverse path to the source), it is ensured that the packet has arrived on the same interface that is listed in the corresponding route entry (so the packet has arrived on the best path back to its origin). If the interface is correct, the packet has arrived from a verifiable source and is legitimate. If a reverse route is not found or the packet arrived on a wrong interface, it is presumed that the packet is spoofed, and it is discarded.

This feature is used for implementing ingress and egress filtering as specified in RFC 2267. It is turned off by default and can be enabled on a specific interface using the following configuration command:

```
ip verify reverse-path interface <interface_name>
```

Ingress filtering is used for checking that outside hosts really have outside addresses, but because the PIX firewall cannot maintain the table of all possible routes on the Internet, most configurations check that packets arriving to the outside interface from the Internet do not have an "inside" source address. Egress

filtering does exactly the opposite: It checks that the packets going to the Internet actually have internal source addresses. This filtering makes tracing any packet back to its origin much easier and prevents most spoofing attacks. Although this can all be accomplished using access lists, the RPF feature provides a much easier and more elegant solution.

Let's consider the following example:

```
PIX1(config)# ip address inside 192.168.1.254 255.255.0.0
PIX1(config)# route inside 192.168.2.0 255.255.255.0 192.168.1.254 1
PIX1(config)# route inside 192.168.3.0 255.255.255.0 192.168.1.254 1
PIX1(config)# ip address outside 1.2.3.1 255.255.255.0 2
PIX1(config)# route outside 0.0.0.0 0.0.0.0 1.2.3.127
PIX1(config)# ip verify reverse-path interface outside
PIX1(config)# ip verify reverse-path interface inside
```

Here, two networks—192.168.2.0/24 and 192.168.3.0/24—are connected to the inside interface, and corresponding entries are created in the routing table. The outside interface has a default route to 1.2.3.127. The RPF feature is enabled on both interfaces. Now, when a packet arrives from the network attached to the inside interface, its source address is checked against the routing table. If this address belongs to one of the two networks 192.168.2.0/24 or 192.168.3.0/24, the route lookup succeeds and the packet is allowed to pass through the firewall. If the address is not from any of these networks, no route will be found, and the packet will be discarded.

If a packet arrives from the Internet to the outside interface, its source is also checked because RPF is active on the outside interface. If this address belongs to one of the networks 192.168.2.0/24 or 192.168.3.0.24, route lookup succeeds, but it is noted that this packet has not arrived on the best path to its origin. (The best path goes through the inside interface.) The packet is obviously a spoofed one and it is dropped. In all other cases, the route lookup also succeeds because there is a default route on the outside interface and the packet is permitted to pass through. Thus *ip verify reverse-path interface inside* provides egress filtering, whereas *ip verify reverse-path interface outside* provides ingress filtering.

If in this configuration we omit RPF verification on the outside interface, only egress filtering on the inside interface will be performed, and spoofed packets from the Internet will be allowed to pass through, whereas any spoofing attempts by inside hosts will be stopped. If RPF verification is enabled only on the outside interface and routes to internal networks are provided, only ingress routing will be performed; outside packets with source IPs belonging to internal networks will be dropped.

NOTE

There are several limitations on using RPF verification. If there is no default route on the outside interface, only the networks mentioned in the routing table are able to send packets to the hosts behind the firewall. Also, do not turn on RPF verification before routing is fully specified, for the same reason. If your network has asymmetric routing, RPF verification will not work correctly.

RPF-related statistics can be viewed with the following command:

```
pix(config)# show ip verify statistics
interface outside: 5 unicast rpf drops
interface inside: 2 unicast rpf drops
```

Counters here show the number of packets dropped by unicast RPF. The number of RPF drops can also be seen in *show interface* results:

```
pix(config)# show interface
interface ethernet0 "outside" is up, line protocol is up
Hardware is i82559 ethernet, address is 00aa.0000.003b
IP address 1.2.3.4, subnet mask 255.255.255.224
MTU 1500 bytes, BW 100000 Kbit half duplex
1183242 packets input, 1222000001 bytes, 0 no buffer
Received 210 broadcasts, 23 runts, 0 giants
4 input errors, 0 CRC, 4 frame, 0 overrun, 0 ignored, 0 abort
1311231 packets output, 565432270 bytes, 0 underruns, 0 unicast rpf drops
0 output errors, 12332 collisions, 0 interface resets
0 babbles, 0 late collisions, 12342 deferred
0 lost carrier, 0 no carrier
input queue (curr/max blocks): hardware (128/128) software (0/1)
output queue (curr/max blocks): hardware (0/2) software (0/1)
```

Line 8 of this output contains a message "0 unicast rpf drops"; this means there were no drops on this interface.

Not all packets are checked with RPF. What actually happens is:

- ICMP packets are all checked because there is no session state for these types of communication.

- TCP and UDP communications have session information maintained by PIX, so only an initial packet is checked against the routing table. All subsequent packets are checked only for the interface they arrived on. This interface should be the interface on which an initial packet arrived.

The following commands delete *ip verify* commands from the configuration and clear packet counts, respectively:

```
clear ip verify reverse-path
clear ip verify statistics
```

Unicast Routing

Configuration of static routing is discussed in Chapter 2. In this section, we describe some more advanced topics related to unicast routing as performed by the PIX firewall.

Static and Connected Routes

You have already learned how to configure static routes on the PIX firewall using the *route* command:

```
route <interface> <ip_address> <netmask> <gateway_address> [<metric>]
```

For example:

```
PIX1(config)# route outside 0.0.0.0 0.0.0.0 1.2.3.4
```

This command configures a static default route on the outside interface to the gateway 1.2.3.4—a default gateway to be used for network traffic. If you issue a *show route* command, the output will include the following line:

```
route outside 0.0.0.0 0.0.0.0 1.2.3.4 1 OTHER static
```

The keyword *OTHER* simply means that this route is a manually entered static route. There is one interesting variation to the *route* command: It is possible to specify an IP address of PIX's own interface instead of a gateway address. This might seem strange from the point of view of the classic static routing, but this is sometimes very useful, especially in a Cisco infrastructure. The PIX itself automatically creates routes of this type when you enter an IP address for an interface.

So, what happens when a route is set to the PIX interface? The simple answer is that the PIX firewall considers the network directly connected and sends an ARP request for the destination address itself instead of requesting for gateway's destination and forwarding the packet to the gateway. The destination host does

not really have to be directly connected; if it is connected via a router that has a *proxy-arp* feature turned on, the router will reply on behalf of the host, the PIX will forward the packet to this router, and the router in turn will forward the packet to the host. Cisco routers and PIX firewalls have proxy ARP turned on by default. For example, if the inside interface has an IP address of 192.168.1.254/24 and two networks, 192.168.2.0/24 and 192.168.3.0/24, are connected to this interface via a router, the following two statements will configure correct routes to these networks (note that the router's IP is not used anywhere; it just has to be in the same network as the inside interface of the PIX):

```
PIX1(config)# route inside 192.168.2.0 255.255.255.0 192.168.1.254
PIX1(config)# route inside 192.168.3.0 255.255.255.0 192.168.1.254
```

The *show route* command displays the corresponding entries in the routing table as:

```
route inside 192.168.1.0 255.255.255.0 192.168.1.254 1 CONNECT static
route inside 192.168.2.0 255.255.255.0 192.168.1.254 1 OTHER static
route inside 192.168.3.0 255.255.255.0 192.168.1.254 1 OTHER static
```

The first entry here was created automatically by the PIX firewall when an IP address was configured on the inside interface. The other two are the result of our two static route entries.

What exactly happens when the default route (outside interface) on the PIX is set to itself? The sequence of steps PIX performs to correctly forward the packet is as follows:

1. The PIX receives a packet on the inside interface destined for the Internet host with IP a.b.c.d.
2. The default route on the outside interface is set to the interface itself. If a separate default gateway was specified, the PIX would simply ARP for the gateway's address and forward the packet there. If not, the PIX sends an ARP request for IP a.b.c.d.
3. Any router (assuming it has proxy ARP turned on) that has a route to a.b.c.d replies with its MAC address on behalf of the host a.b.c.d.
4. The PIX forwards the packet to this router, which will handle it from there.
5. The PIX also adds an entry to its ARP table for IP address a.b.c.d with the MAC address of the router.

The PIX firewall also has the proxy ARP feature turned on by default, so it can act in the same way as the router in the previous example. It is possible to turn the feature off on a specific interface using:

```
sysopt noproxyarp <interface>
```

Configuring & Implementing…

Proxy ARP and One-Armed Routing Mode

In case you have not heard the phrase, *"one-armed" routing* means that the router has only one interface (with more than one IP address on it). All it does is receive a packet from the network and redirect it to another router/host on the same LAN but maybe on another IP network. This is sometimes useful, but PIX cannot do this, because its Adaptive Security Algorithm does not allow any packet to exit on the same interface as it arrived.

Combined with the default proxy ARP feature, this feature can play tricks on your routing. For example, if a router is behind an inside interface and some host sends an ARP request for this router's IP, PIX will reply instead (or together with the router) and the packet is forwarded to the PIX. Here comes the problem: The packet needs to be forwarded to the real router, but PIX cannot do this; the packet cannot exit on the same interface.

So, if you prefer to completely control your static routing and you have created all static routes with correct gateways, it is always better to turn off proxy ARP on all interfaces; it has a nasty habit of getting in the way.

Routing Information Protocol

Beside static routes, the PIX firewall also supports Routing Information Protocol (RIP) versions 1 and 2. This protocol is the simplest dynamic routing protocol and is described in RFCs 1058, 1388, and 2082. Roughly speaking, a router broadcasts (or it may use multicast in version 2) its entire routing table to its neighbors, and they update their tables.

Each PIX interface can be configured either to broadcast (multicast) itself as a default route for the network or to passively listen for routing updates from other

routers on the LAN. The simple syntax of the RIP configuration command is as follows:

```
rip <if_name> [default | passive] version [1 | 2]
```

The *default* and *passive* keywords define the mode RIP runs on the interface *if_name*. The *default* parameter specifies that a default route should be advertised, and *passive* means listening for updates from other routers. The *version* parameter specifies the version of RIP to use on the interface. If a version is not specified, version 1 is assumed. The major differences between RIPv1 and RIPv2 are that RIPv2 can use multicast to the address 224.0.0.9 instead of broadcasts and that it can use authentication. RIPv1 uses broadcasts only and no authentication of updates. RIPv2 is also a classless routing protocol, which means that it can exchange routing information for networks such as 172.16.1.0/24, whereas RIP v1 uses only networks of A, B, and C classes—for example, Class B network 171.16.0.0/16. Generally, it is better to use RIPv2 if there is no need to interact with older RIPv1 devices.

NOTE

Before PIX version 5.3, the PIX firewall was capable of using only broadcasts for RIPv2. Versions 5.3 and later use multicast to the address 224.0.0.9. By default, when you use RIPv2 on the PIX, it sends updates to 224.0.0.9. If passive mode is configured with RIPv2, the PIX accepts multicast updates with the address of 224.0.0.9, and this multicast address is registered on the corresponding interface. Only Intel 10/100 and Gigabit interfaces support multicasting. When RIP configuration commands are removed from the configuration, this multicast address is unregistered from the interface.

If you have a router that talks multicast RIPv2 to an older PIX (before version 5.3), the PIX will not receive any updates. It is possible to switch the router into unicast mode using a command *neighbor <pix_address>* in its RIP configuration section. The PIX is capable of receiving unicast updates in any version that supports RIP.

Here is an example of RIP v1 configuration:

```
PIX1(config)# show rip
rip outside passive
no rip outside default
```

```
rip inside passive
no rip inside default
PIX1(config)# rip inside default
PIX1(config)# show rip
rip outside passive
no rip outside default
rip inside passive
rip inside default
```

The first *show rip* command displays the default state of configuration: all interfaces listen passively. Then the inside interface is configured to broadcast itself as a default route. Note that the passive listening mode was not turned off by this mode; you would need to disable it separately with *no rip inside passive* if you wanted to turn it off.

RIP v2 also supports two types of authentication: cleartext passwords and MD5 hashes. This feature of RIPv2 protocol adds one more field to the transmitted routing update—an authentication field. It can contain either a cleartext password (not recommended) or a keyed MD5 hash of the whole message. *Keyed* means that there is a key that is used to compute a hash value of the message. PIX configuration is very simple in both cases: An extra parameter needs to be added to the basic configuration command:

```
rip <if_name> [default | passive] version 2 authentication [text | md5]
    <key_string> <key_id>
```

For example, the following command uses a cleartext password of *mysecretkey* while broadcasting the default gateway on the inside interface:

```
rip inside default version 2 authentication text mysecretkey 1
```

The following command lists only the messages with a correct MD5 hash keyed by a key *anothersecretkey*:

```
rip outside passive version 2 authentication md5 anothersecretkey 2
```

The *key_id* parameter (a number at the end of the line) is a key identification value and must be the same on all routers with which the PIX communicates.

RIP authentication on routers is more complicated. You need to set up a key chain with some keys (these keys are numbered and are exactly the *key_id* you need to provide in configuring PIX) and turn the authentication on. A sample partial router configuration corresponding to our case of MD5 authentication is:

```
interface ethernet 0
 ip rip authentication key-chain mykeys
 ip rip authentication mode md5
!
router rip
 network 172.16.0.0
 version 2
!
key chain mykeys
 key 2
 key-string anothersecretkey
```

NOTE

The PIX firewall is able to support one and only one key ID per interface. Keys have unlimited lifetimes, and it is recommended that you change them every two weeks or so. Note also that if you use Telnet to configure these keys, they might be exposed.

The *clear rip* configuration mode command removes all RIP configuration statements from the PIX firewall.

Stub Multicast Routing

IP multicasting is becoming increasingly popular, especially in SOHO environments, where hosts are connected via fast links. Multicasting was introduced as a method of packet delivery to multiple hosts. In broadcasting, each host receives all packets sent by a server. In multicasting, a host must join one or more *multicast groups*, represented by a specific IP address (these addresses are 224.0.0.0-239.255.255.255) and then it will listen only for packets destined for this group. Of course, the nature of broadcasting and multicasting implies that it can be used only for UDP transmission, because TCP always requires two endpoints.

So how exactly does multicasting work? As noted, there is a set of multicast group addresses (Class D IP addresses, 224.0.0.0 through 239.255.255.255). A group of hosts listening to a particular multicast group address is called a *host group*. A host group is not limited to one network and can include hosts from many

networks at the same time. Membership in a group is dynamic; hosts can enter and leave a group at will. The number of hosts in a group is not limited, and a host does not have to be a member of the group to send a message to this group.

When a host sends a message to a specific group address, this address is not subject to the ARP resolution process. It is simply converted into an Ethernet address by special rules, and an Ethernet frame is sent out with the resulting destination MAC address. If all recipients are on the same physical network, everything else is very simple: Listening hosts decide if the packet is sent to them by looking at the MAC address and its correspondence with the group addresses they are listening on. But multicast groups are not limited to one network by definition, so there is a need for some means of passing these messages through routers and a means of informing routers if there are any hosts from a specific multicast group on a given physical network. This is done using Internet Group Management Protocol (IGMP).

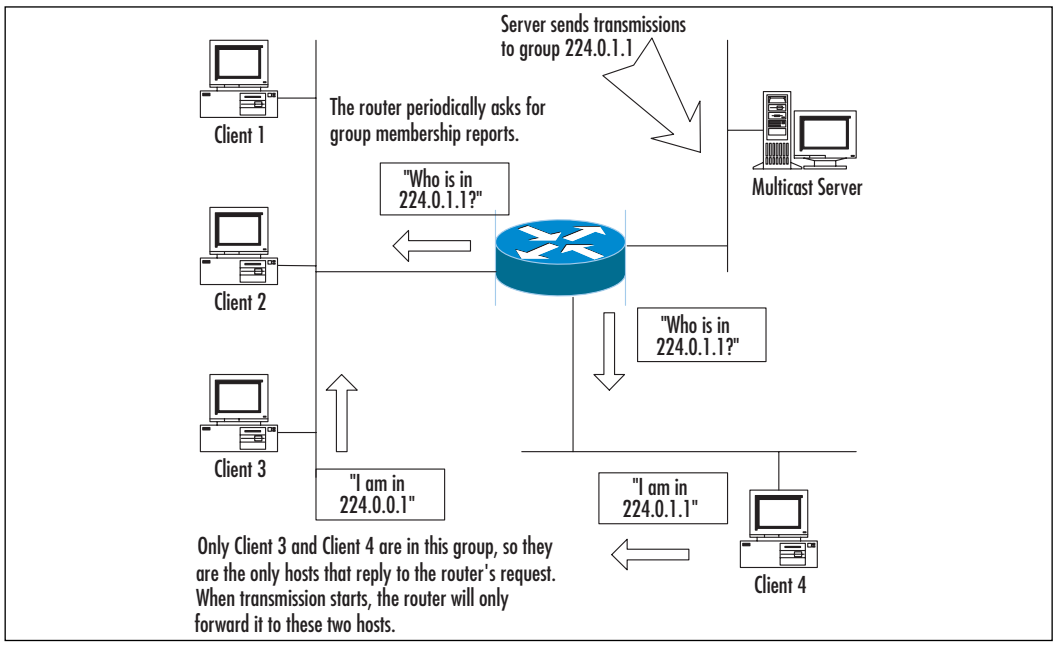
IGMP is similar to ICMP in that it is also considered part of the IP layer. It is IP protocol number 2. Its basic functionality is as follows:

- When a host joins a multicast group, it informs the router by sending it an IGMP message.
- When a host leaves the group, it does not send any reports about this event (see the next two points).
- A multicast router regularly sends IGMP requests out each of its interfaces requesting connected hosts to report to the multicast groups to which they belong.
- A host responds to the request by sending one IGMP report for each group to which it belongs.

Figure 4.13 illustrates this IGMP exchange.

Since version 6.2, the PIX can process multicast and IGMP messages. It does not have full capabilities of a multicast router, but it can act as a "stub router" or IGMP proxy agent. An IGMP proxy agent is a device that is able to forward IGMP requests and replies between multicast routers and hosts. When the source and destination of multicast transmissions are divided by a PIX firewall, two obvious cases are possible: when the source of a transmission (or a multicast router) is on a lower security-level interface than the destination and when the source (router) is on a higher security-level interface than the destination. Let's look at these two cases separately.

Figure 4.13 IGMP Used to Report Membership in a Multicast Group



SMR Configuration with Clients on a More Secure Interface

In this case, a multicast router and a server are on the outside interface of the PIX firewall, and clients are on the inside. The PIX needs to be able to pass multicast traffic from the server and IGMP requests from the router to the inside hosts. It also needs to pass IGMP messages from the internal hosts to the outside router.

All SMR configurations start with the following configuration mode command:

```
multicast interface <interface> [max-groups <number>]
```

This command enables multicast features on the specified interface. The interface is placed into multicast promiscuous mode, and it enters a submode of multicast configuration for a specific interface. (This is a rare case with the PIX because there are very few submodes in configuration mode.) An optional *max-groups* parameter defines the number of multicast groups that can appear on the interface at any given time. The default setting is 500; the number can be up to 2000. This mode has subcommands like this:

```
igmp <command>
```

NOTE

To set the version of IGMP used, use the *igmp version* {*1* | *2*} sub-command under the *multicast* command.

In our case, the PIX needs at least to be able to receive multicast transmissions on its outside interface, so we need to configure:

```
PIX(config)# multicast interface outside
```

Actually, there is not much more to configure on the outside interface. We can optionally configure some counters and protocol options or access control, but this is not specific for a case and is described later. After exiting this multicast configuration mode (but while we're still in configuration mode), we need to configure multicast on the inside interface:

```
PIX1(config)# multicast interface inside
```

The inside interface needs some more configuration. After we enter this mode, we need to configure the interface to which the PIX should forward all IGMP messages from clients. This is the less secure interface where the router is located:

```
PIX1(config-multicast)# igmp forward interface outside
```

Don't forget that this command is entered while we are in the interface multicast configuration mode. *Outside* is the interface name to forward IGMP messages to from the interface being configured. If you have a multicast router on an interface named *dmz1*, the command will look like:

```
PIX1(config-multicast)# igmp forward dmz1
```

If any clients on the inside network are not IGMP-capable, but we still want them to receive multicast traffic from some group, we need to configure the inside interface to join this multicast group statically with the command:

```
igmp join-group <multicast_address>
```

For example:

```
PIX1(config-multicast)# igmp join-group 224.1.1.1
```

With this interface configured, the PIX outside interface acts as a host interested in receiving transmissions for this group, and then the received data will be forwarded to the inside network. Here is an example of the simplest multicast configuration:

```
PIX1(config)# multicast interface outside
PIX1(config-multicast)# exit
PIX1(config)# multicast interface inside
PIX1(config-multicast)# igmp forward interface outside
```

Here is a more complicated example with non-IGMP capable multicast clients who want to receive transmissions for group 224.10.0.9:

```
PIX1(config)# multicast interface outside
PIX1(config-multicast)# exit
PIX1(config)# multicast interface inside
PIX1(config-multicast)# igmp forward interface outside
PIX1(config-multicast)# igmp join-group 224.10.0.9
```

Clients on two interfaces, *inside* and *dmz*:

```
PIX1(config)# multicast interface outside
PIX1(config-multicast)# exit
PIX1(config)# multicast interface inside
PIX1(config-multicast)# igmp forward interface outside
PIX1(config-multicast)# exit
PIX1(config)# multicast interface dmz
PIX1(config-multicast)# igmp forward interface outside
```

SMR Configuration with Clients on a Less Secure Interface

This case is simpler. All you need to do is enable multicast processing on both interfaces and create static multicast routes for passing traffic between the clients and the servers (and routers). Multicast processing is enabled with:

```
PIX1(config)# multicast interface outside
PIX1(config-multicast)# exit
PIX1(config)# multicast interface inside
```

Multicast route are created using the *mroute* command (which is not a sub-command of the multicast command):

```
mroute <src> <srcmask> <in-if-name> <dst> <dstmask> <out-if-name>
```

The *src* and *srcmask* parameters are the IP address and subnet mask of a multicast source host/router (just normal IP addresses, not multicast addresses.). The *in-if-name* parameter specifies the interface connected to the source. *dst* and *dstmask*

are the multicast group address and subnet mask to which the server is sending its transmission. Finally, *out-if-name* is the interface connected to the multicast clients. For example:

```
PIX1(config)# mroute 192.168.2.25 255.255.255.255 inside 224.0.1.1 255.
    255.255.255 outside
```

Here is an example configuration in the case of two servers: 192.168.2.25 on the inside interface multicasting to group 224.1.1.1 and 10.2.3.4 on the dmz interface multicasting to the group 230.1.1.1 and no internal clients:

```
PIX1(config)# multicast interface outside
PIX1(config-multicast)# exit
PIX1(config)# multicast interface inside
PIX1(config-multicast)# exit
PIX1(config)# multicast interface dmz1
PIX1(config-multicast)# exit
PIX1(config)# mroute 192.168.2.25 255.255.255.255 inside 224.1.1.1 255.
    255.255.255 outside
PIX1(config)# mroute 10.2.3.4 255.255.255.255 dmz 230.1.1.1 255.255.255.
    255 outside
```

Access Control and Other Options

It is possible to restrict access to multicast transmissions using the usual PIX means: access lists. In the preceding case with hosts on the inside interface, we could restrict the groups from which the internal hosts can receive transmissions. For example, to allow only multicast transmissions to a group address 224.1.1.1, you should create an access list similar to this:

```
PXI1(config)# access-list 10 permit igmp any 224.1.1.1 255.255.255.255
```

Then apply it to the outside interface:

```
PIX1(config)# multicast interface outside
PIX1(config-multicast)# igmp access-group 10
```

Now only IGMP polls for group 224.1.1.1 will be able to pass through PIX, and thus only members of this group will be known to a multicast router. This prevents the router from sending traffic destined for any other group address in this direction.

Other subcommands of the *multicast* command include:

```
igmp query-interval <seconds>
```

This command sets the interval at which IGMP messages will be sent out this interface. The default interval is 60 seconds. The maximum timeout for response (for IGMP version 2 only) can be set using:

```
igmp query-max-response-time <seconds>
```

The default setting is 10 seconds.

Configured settings can be cleared using corresponding *clear* commands. The following command clears the IGMP cache either for a specific group address or the whole cache on the specified interface:

```
clear igmp group [<group-addr> | interface <interface-name>]
```

The following command clears multicast routes for specified transmission source, for a group address, or all routes on the interface:

```
clear mroute [<src-addr> | <group-addr> | interface <interface-name>]
```

Another set of commands allows viewing of multicast configuration for the interface, multicast group, routes, and so on:

```
show igmp
show multicast [interface <interface-name>]
show igmp group [grou<p-addr> | interface <interface-name>]
show mroute [<src-addr> | <group-addr> | interface <interface-name>]
```

An example output of the *show igmp* command is:

```
pix(config)# show igmp
IGMP is enabled on interface inside
Current IGMP version is 2
IGMP query interval is 60 seconds
IGMP query timeout is 125 seconds
IGMP max query response time is 10 seconds
Last member query response interval is 1 seconds
Inbound IGMP access group is
IGMP activity: 0 joins, 0 leaves
IGMP querying router is 10.0.1.1 (this system)
IGMP Connected Group Membership
Group Address Interface Uptime Expires Last Reported
```

Two *debug* commands allow monitoring of multicast-related events. This command monitors all IGMP messages passing through the PIX:

```
debug igmp
```

The following command monitors all events related to multicast forwarding:

```
debug mfwd
```

PPPoE

Point-to-Point Protocol over Ethernet (PPPoE), documented in RFC 2516, is an encapsulation of Point-to-Point Protocol (PPP, RFC 1661) for Ethernet networks (which include DSL modems and cable connections). PPPoE is often used in SOHO environments because it allows ISPs to use their existing remote access infrastructure and, as its most important feature, allows authenticated IP address assignment. PPPoE links are established in two main phases:

- **Active discovery phase** During this first phase, a PPPoE client attempts discovery of the PPPoE server, also called the *address concentrator* (AC). The PPPoE layer is established and a session ID is assigned.
- **PPP session phase** A PPP link is established (encapsulated in Ethernet) by the usual means: options and link layer protocols are negotiated etc. PPP authentication (PAP, CHAP, or MS-CHAP) is performed.

After the session is established, data travels between endpoints encapsulated in PPPoE headers.

The PIX firewall supports PPPoE since software version 6.2. Most of the PPPoE configuration is performed using the *vpdn* command. PPPoE configuration starts with configuring the username and password to be used by the PIX in establishing a link to the server.

NOTE

The PIX only supports PPPoE client functionality. PPPoE clients can be enabled only on the outside interface at this time (version 6.2).

First, a VPDN group needs to be created:

```
vpdn group <group_name> request dialout pppoe
```

The *group_name* parameter can be anything you like. It is used to group all PPPoE settings together. For example:

```
PIX1(config)# vpdn group my-pppoe-group request dialout pppoe
```

Then the authentication type needs to be selected (if required by an ISP):

```
vpdn group <group_name ppp> authentication pap | chap | mschap
```

PAP is Password Authentication Protocol, CHAP is Challenge-Handshake Authentication Protocol, and MS-CHAP is Microsoft's version of CHAP. With the same group name, this command selects an authentication protocol for this specific PPPoE group—for example, with CHAP authentication:

```
PIX1(config)# vpdn group my-pppoe-group ppp authentication chap
```

Your ISP assigns the username and password to your system, and they are configured on PIX with the following commands:

```
vpdn group <group_name> localname <username>
vpdn username <username> password <pass>
```

The second of these commands associates a username with the password, and the first command assigns the username to be used for a specific group, for example:

```
PIX1(config)# vpdn group my-ppoe-group localname witt
PIX1(config)# vpdn username witt password cruelmail
```

These commands assign the username *witt* and password *cruelmail* to be used for the PPPoE dialout group *my-pppoe-group*. After configuring authentication, the next task is to enable the PPPoE client on the PIX. This is done in the configuration of the outside interface:

```
ip address outside pppoe [setroute]
```

After this command is entered, the current PPPoE session is terminated and a new one is established. The *setroute* parameter allows automatically setting the default route for the outside interface. The MTU on the outside interface is automatically set to 1492, which is the correct setting to provide PPPoE encapsulation. It is also possible to designate a fixed IP address for the outside interface. The PIX still has to provide the ISP with the correct username and password in order to establish the session:

```
PIX1(config)# ip address outside 1.2.3.4 255.255.255.0 pppoe
```

It is possible to use the *dhcp auto_config* command if you run the DHCP server on PIX in order to pick up DNS and WINS settings from your provider via the PPPoE client:

```
PIX1(config)# dhcpd auto_config outside
```

To monitor and troubleshoot the PPPoE client, use the following commands:

```
show ip address outside pppoe
debug pppoe event | error | packet
show vpdn session pppoe [id <sess_id>|packets|state|window]
```

Examples of output are as follows:

```
PIX1(config)# show vpdn
Tunnel id 0, 1 active sessions
time since change 10240 secs
Remote Internet Address 10.0.1.1
Local Internet Address 192.168.2.254
1006 packets sent, 1236 received, 98761 bytes sent, 123765 received
Remote Internet Address is 10.0.1.1
Session state is SESSION_UP
Time since event change 10237 secs, interface outside
PPP interface id is 1
1006 packets sent, 1236 received, 98761 bytes sent, 123765 received
PIX1(config)# show vpdn tunnel
PPPoE Tunnel Information (Total tunnels=1 sessions=1)
Tunnel id 0, 1 active sessions
time since change 10240 secs
Remote Internet Address 10.0.1.1
Local Internet Address 192.168.2.254
1006 packets sent, 1236 received, 98761 bytes sent, 123765 received
PIX1(config)# show vpdn session
PPPoE Session Information (Total tunnels=1 sessions=1)
Remote Internet Address is 10.0.1.1
Session state is SESSION_UP
Time since event change 100238 secs, interface outside
PPP interface id is 1
1006 packets sent, 1236 received, 98761 bytes sent, 123765 received
```

Summary

The Cisco PIX firewall is an advanced product and has many different options for supporting various application-layer protocols as well as protecting against network-layer attacks. It also supports content filtering for outbound Web access, intrusion detection, various routing options such as RIP and stub multicast routing, and DHCP server and client functionality.

Many protocols embed extra IP address information inside the exchanged packets or negotiate additional connections on nonfixed ports in order to function properly. These functions are handled by the PIX application inspection feature (also known as *fixup*). PIX supports FTP clients and servers in active and passive modes, DNS, RSH, RPC, SQL*Net, and LDAP protocols. It also supports various streaming protocols such as Real-Time Streaming Protocol, NetShow, and VDO Live. Another set of supported protocols includes all H.323, SCCP, and SIP—all used in VoIP applications. The PIX monitors passing packets for the embedded information and updates its tables or permits embryonic connections according to this information. It is also able to NAT these embedded addresses in several cases.

Content filtering features on the PIX can be used to enforce a company's acceptable use policy. The PIX can interface with Websense (www.websense.com) or N2H2 (www.n2h2.com) servers and deny or allow internal clients access specific Web sites. The PIX is also able to filter out Java applets and ActiveX code from incoming Web pages to protect clients against malicious code.

The PIX firewall supports the same set of atomic intrusion detection signatures as the Cisco IOS firewall. This set is a subset of signatures supported by the Cisco Secure IDS product. These signatures are divided into two sets: informational and attack. It is possible to configure different response options for each set of signatures. The responses range from simple alerting via syslog to blocking the connection in which a signature was detected.

For SOHO environments, the PIX firewall provides DHCP server and client functionality, although server capabilities are rather limited. DHCP server supports a couple of specific options that are used by Cisco IP Phones. Other useful PIX features include support of stub multicast routing and PPP over Ethernet client capabilities. It also supports RIP versions 1 and 2, including authentication and multicast updates for version 2.

Finally, the PIX has embedded protection against various DoS attacks, such as SYN floods, attacks on AAA mechanisms, and excessive fragmentation. Antispoofing is supported by the reverse-path forwarding feature.

Solutions Fast Track

Handling Advanced Protocols

- ☑ Many applications use more than one connection to operate; only one of these connections occurs on a well-known port, whereas others use dynamically assigned port numbers, which are negotiated in the process of communication. This makes firewalling by means of access lists very difficult. The PIX supports application inspection for many such protocols, which allows it to operate correctly with them.
- ☑ The main command used to configure application inspection is the *fixup* command. It can be used for simpler protocols such as FTP, SMTP, or RSH.
- ☑ Newer versions of the PIX firewall offer support for various VoIP protocols, such as H.323, SCCP, and SIP.

Filtering Web Traffic

- ☑ Filtering Web traffic can be useful in two main cases. The first is if you want to use your firewall to enforce security policies such as an acceptable use policy, which may specify that internal users cannot use the company's Internet connection to browse certain categories of Web sites. The second is to protect internal users from malicious Web servers that embed these executable applets in their Web pages, because such executable content can contain viruses or Trojan horses.
- ☑ The PIX supports two types of content filtering servers: Websense and N2H2. The main commands for configuring this feature are *filter-url* and *url-server*. The PIX also provides many commands for monitoring and tuning the filtering process.
- ☑ Active code filtering is limited to stripping <*object*> and <*applet*> tags from the source of inbound Web pages. This stripping can only occur when opening and closing tags are contained in the same IP packet. This filtering is configured with the *filter java* and *filter activex* commands.

Configuring Intrusion Detection

- ☑ The PIX supports a limited embedded set of (over 55) IDS signatures. These are signatures that can be detected by examining a single packet and do not require any session information. This set can be updated only by upgrading the PIX software.
- ☑ The signatures are divided into two sets: informational and attack. It is possible to configure different reaction options for each set—syslog alarm, dropping the packet, or dropping the whole connection in which the attack has occurred.
- ☑ Any signature can be disabled so that it will no longer be detected. This change has a global effect; this signature will not be detected on any interface by any audit until the signature is enabled again.

DHCP Functionality

- ☑ The Cisco PIX firewall can act both as a DHCP server and a client. PIX DHCP features are best suited for small networks because they have some limitations—for example, a DHCP server can support a maximum of 256 clients. There is also no BOOTP support and no failover support.
- ☑ The DHCP client can be configured only on the outside interface. It is able to obtain an IP address, subnet mask, default route, and DNS and WINS settings from the server. The obtained address can be used for NAT or PAT on the outside interface.
- ☑ The DHCP server can be configured only on the inside interface and serves only directly connected clients. The number of active clients is dependent on the PIX model and software version. It is possible to pass some settings that are obtained by PIX DHCP clients from the outside interface to the DHCP server running on the inside interface.

Other Advanced Features

- ☑ The PIX has built-in protection against DoS attacks such as SYN floods and AAA resource exhaustion. It also supports virtual reassembly of IP fragments and can impose some extra limitations on fragmented traffic.

- ☑ The PIX supports antispoofing protection using reverse-path forwarding (RPF). It also supports advanced routing features such as dynamic routing using RIP versions 1 and 2 and stub multicast routing.
- ☑ The PIX firewall can act as a PPPoE client on DSL or cable connections.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: What happens when FTP fixup is not enabled?

A: There are several cases:

- Outbound active FTP sessions will not work because the outside servers will not be able to open a data channel to an inside client.
- Outbound passive FTP sessions will work normally if outbound traffic is not explicitly disabled, because all connections in this case are initiated by an inside client.
- Inbound FTP active connections will work normally if there are a static NAT entry and an access list allowing outside clients to connect to the inside server.
- Inbound FTP passive FTP connections will not work because outside clients will not be able to open data connections to the inside server.

Q: I have a PIX and an SMTP server configured on its inside network. Sometimes I get two copies of incoming mail messages. What is wrong with my server?

A: Nothing is wrong; there is a slight misbehavior on the PIX side. You probably have *fixup protocol smtp* configured. Some versions of PIX software send an error message to relaying servers when a final dot in the message body and *<CR><LF>* are not in the same IP packet. In this case, your internal server accepts the message for delivery, but the outside relaying server treats this as

an error and attempts delivery again. Most of the time, this condition does not happen twice in a row, so the second time delivery goes without error and you receive two copies of the same message. If this really irritates you, either turn SMTP fixup off or upgrade the PIX software.

Q: Is it possible to filter e-mail content in any way similar to Web content filtering?

A: No, this is not possible. The PIX does not inspect the contents of TCP packets related to e-mail and currently does not support any outside filtering servers.

Q: I have two links to my ISP, and I turned on RPF. Now half my traffic is being denied by the PIX. What should I do?

A: The only solution here is to turn RPF verification off. It simply does not work in a situation with asymmetric routing, where a reply to the packet may come on a path other than the packet itself.

Q: I cannot get NFS to work through the PIX, although I configured an access list that permits clients access to the portmapper on the server.

A: You are probably using NFS over TCP. The PIX does not support application inspection for RPC connections over TCP. Reconfigure your server to use UDP only.

Chapter 5

Configuring Authentication, Authorization, and Accounting

Solutions in this chapter:

- AAA Concepts
- Cisco Secure ACS for Windows
- Configuring Console Authentication
- Configuring Command Authorization
- Configuring Authentication for Traffic Through the Firewall
- Configuring Authorization for Traffic Through the Firewall
- Configuring Accounting for Traffic Through the Firewall
- Configuring Downloadable Access Lists

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

The use of user-level security is becoming increasingly popular. This type of security enables us to develop and enforce policies on a per-user basis. Seldom is a network designed to be open to *all* people or *no* people. Generally, you want to provide access to some people and not to others. For example, a server holding sensitive salary information should be accessible to certain members of the Human Resources department and no one else. How do you confirm that the person accessing the data is authorized to do so? This granular level of administration based on user or group name is possible using *authentication, authorization, and accounting (AAA).* In this chapter, you will learn how to use and configure AAA on the Cisco PIX firewall. You will also learn about the RADIUS and TACACS+ security protocols and the advantages and disadvantages of using each one.

The PIX firewall is capable of acting as an AAA client. The PIX can provide AAA functionality for administrative access to the firewall itself, as well as for traffic passing through the firewall. In this chapter, you will learn how to use this functionality with Cisco Secure Access Control Server for Windows, Cisco's AAA server.

AAA Concepts

AAA is an architectural framework for providing the independent but related functions of authentication, authorization, and accounting, which are defined as follows:

- *Authentication* is the process of identifying and validating a user before allowing access to network devices and services. User identification and authentication are critical for the accuracy of the authorization and accounting functions.
- *Authorization* is the process of determining a user's privileges and access rights after they have been authenticated.
- *Accounting* is the process of recording user activities for accountability, billing, auditing, or reporting purposes.

The AAA framework typically consists of a client and a server. The AAA client (typically a router, NAS, or firewall) requests authentication, authorization, and/or accounting services from an AAA server (typically a UNIX or Windows server with appropriate software) that either maintains databases containing the

relevant AAA information locally or communicates with an external database that contains the information. Examples of external databases are a Windows NT domain, Active Directory, LDAP, an SQL Server database, and the UNIX password database. Here are some typical conditions under which using an AAA framework would be effective:

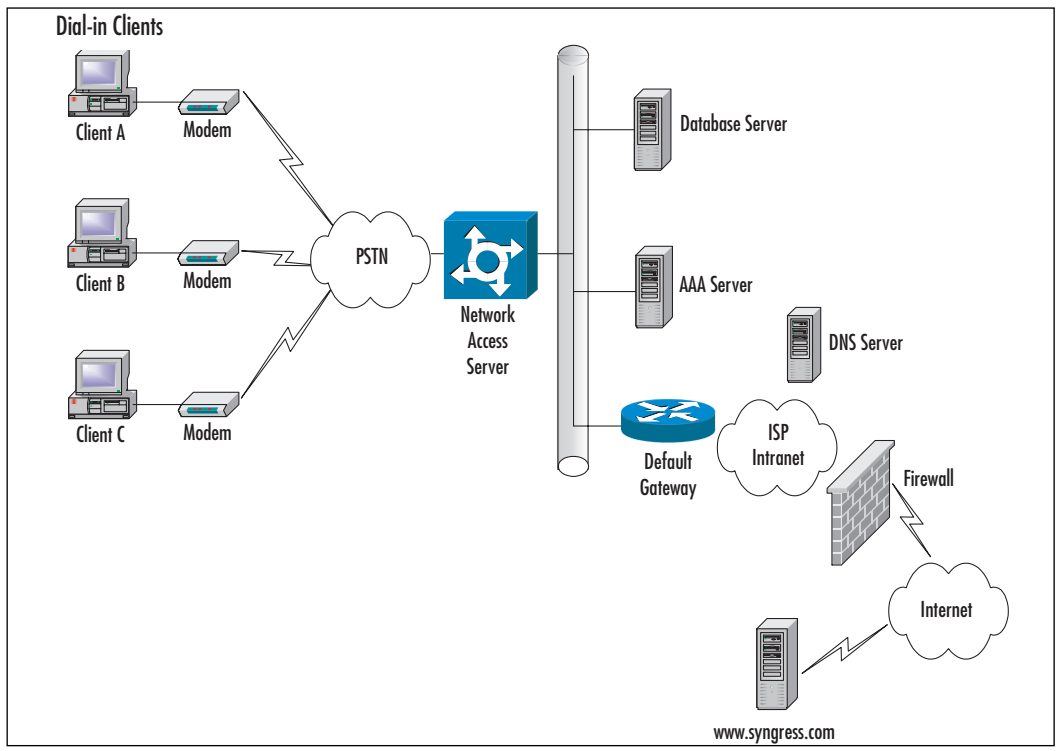
- **To provide centralized authentication for the administration of a large number of firewalls** An example is a small to medium-sized business that has a relatively high ratio of firewalls to security administrators. Centralized authentication would ease the administrative burden, but because the number of administrators is low, centralized authorization and accounting might not be beneficial.
- **To provide flexible authorization capabilities** An example is a global enterprise that has a large number of firewalls and many administrators. Administrative duties might be divided along operational and configuration lines such that the implementation of centralized authorization would be an effective addition to centralization authentication.
- **To provide relevant usage or billing information** An example is a service provider that charges customers based on network usage statistics. In this case, the centralized authentication and authorization would be an effective means of supporting firewall administration, while centralized accounting would provide the business with network usage information for billing.

Examples of AAA happen in everyday life outside of computers and Cisco devices. For example, when you go to an automatic teller machine (ATM) to withdraw money, you must first insert your bankcard and enter your personal identification number (PIN). At this point, you are authenticating yourself as someone who has the authority to withdraw money from this account. If your card and PIN are both valid, you will successfully authenticate and can continue the task of withdrawing money. If you have entered an incorrect PIN or your card has been damaged (or stolen) and the criteria cannot be validated, you will not be able to continue. Once authenticated, you will be permitted to perform certain actions, such as withdraw, deposit, or check your balance on various accounts. Based on your identity (your bank card and your PIN), you have been pre-authorized to perform certain functions, such as withdrawing your hard-earned money. Finally, once you complete the tasks you are authorized to perform, you will be provided with a statement describing your transactions as well

as the remaining balance in your account. The bank will also record your transactions (probably more verbosely than what is on your statement) for accounting purposes.

Now let's look at an example of the same principle applied to a Web site. In Figure 5.1, Client A is attempting to access the Web site www.syngress.com. In order to accomplish this goal, Client A must first connect to its local Internet service provider (ISP) to gain access to the Internet. When Client A connects to the ISP, it is prompted for a set of logon credentials (authentication) by the network access server, or NAS, before it can fully access the Internet.

Figure 5.1 Implementation of AAA at an ISP



An NAS is a device that provides access to a target network (for example, an Internet, corporate network) and usually has an interface connected to the target network and one or more interfaces connected to an external network (such as the Internet or the public switched telephone network, or PSTN). It receives connections from clients on the external interface and provides access to the target network. A security server is typically a device such as a Windows NT or UNIX server that is running TACACS+, RADIUS, or another service that

enforces security. In Figure 5.1, the AAA server is an example of a security server. Once the client has entered its credentials and the AAA server has validated them, if the security policy permits it to use the Internet (authorization), it can now connect to the desired Web site (www.syngress.com). As a policy, the ISP has decided to log all customer connections to the AAA server (accounting). This example illustrates all three elements of AAA: authentication, authorization, and accounting.

NOTE

Do not be confused about AAA terminology. In the example shown in Figure 5.1, the AAA client is the NAS, not the PCs that are dialing up through modems.

Authentication

Authentication is the process of identifying and validating a user. This process typically relies on one or more of the following general methods:

- **Something the user knows** This approach is authentication by knowledge, where the identity is verified by something known only by the user. This is the most common and the weakest approach used for authentication today. Examples include both the UNIX and Windows NT/2000 login process, in which the user is typically prompted to enter a password. The integrity of this authentication process depends on the "something" being both a secret and also hard to guess—a dual goal that is not easily ensured. Some organizations have extended the UNIX and Windows NT/2000 login process to require tokens or smart cards (something you have), or biometrics (something you are), other authentication methods discussed in the points that follow.
- **Something the user possesses** This approach is authentication by possession, where the identity is verified by something possessed only by the user. This authentication approach is becoming more common and is used in most people's daily lives in the form of keys and security badges. The integrity of this authentication process depends on the "something" being unique and possessed only by the user, such as a smart card. If this object is lost or stolen, the authentication process is compromised.

- **Something the user is** This approach is authentication by user characteristic, where the identity is verified by something that is unique *about* the user. This is known as the field of biometrics. Many products are currently being developed and produced that use techniques such as fingerprint scans, retina scans, and voice analysis. ATMs are beginning to be deployed with biometric authentication. This is the strongest approach to authentication and avoids the common problems with the other approaches (such as a password being guessed or a card being lost or stolen). However, this approach is also the most difficult to implement.

Two-factor authentication uses a combination of two of the preceding approaches to authenticate user identities. Typically, two-factor authentication is a combination of something the user possesses and something the user knows. A common example is the use of an ATM card (something possessed) and an associated PIN (something known) to access an account via an ATM machine. In the computer world, you can find two-factor authentication in the form of tokens, where a combination of a PIN plus a changing value on the token is used for authentication.

Within the AAA framework, authentication occurs when an AAA client passes appropriate user credentials to the AAA server and requests that the server authenticate the user. The AAA server attempts to validate the credentials, and responds with either an "accept" or a "deny" message. AAA authentication is typically used in the following scenarios:

- To control access to a network device such as a router, NAS, or firewall
- To control access to network resources through a network device such as a router, NAS, or firewall

Authorization

Authorization can be described as the act of permitting predefined rights or privileges to a user, a group of users, a system, or a process. Within the AAA framework, a client will query the AAA server to determine the actions a user is authorized to perform. The AAA server will return a set of attribute-value (AV) pairs that defines the user's authorization. The client is then responsible for enforcing user access control based on those AV pairs. AAA authorization is typically used in the following scenarios:

- To provide authorization for actions attempted while logged into a network device.
- To provide authorization for attempts to use network services through a network device.

Accounting

Accounting is a method which records (or accounts) who, what, when, and where an action has taken place. Accounting enables you to track both the services that users are accessing and the amount of resources that they are consuming. This data can later be used for accountability, network management, billing, auditing, and reporting purposes. Within the AAA framework, the client sends accounting records that consist of accounting AV pairs to the AAA server for centralized storage.

AAA Protocols

The previous sections provided a high-level overview of AAA and the benefits of using it. This section describes how to implement AAA services on Cisco network devices.

You can configure most Cisco devices, including routers, access servers, firewalls, and virtual private network (VPN) devices, to act as AAA clients. You can configure these network devices to request AAA services to protect the devices themselves from unauthorized access. You can also configure them to request AAA services to protect the network from unauthorized access by users attempting to use the devices as an access point.

RADIUS

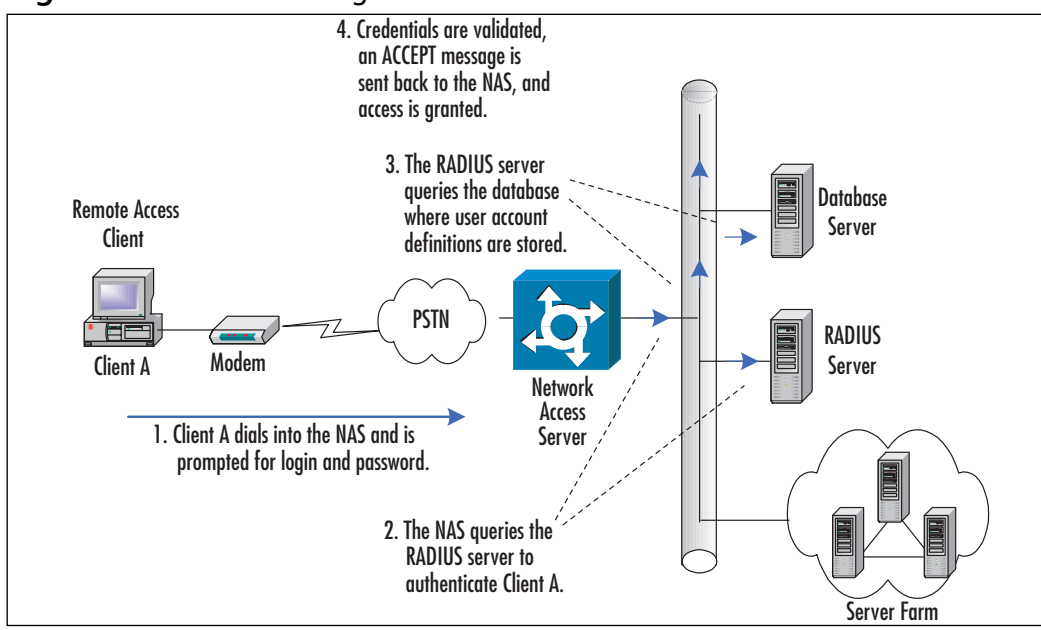
The Remote Access Dial In User Service (RADIUS) protocol was developed by Livingston Enterprises, Inc., as an access server authentication and accounting protocol. Although many RFCs are available on RADIUS, the main specification can be found in RFC 2058, which was made obsolete by RFC 2865. The RADIUS accounting standard is documented in RFC 2059, which was made obsolete by RFC 2866.

RADIUS can be used as a security protocol for a network of any size, from large enterprise networks such as ISPs to small networks consisting of a few users requiring remote access. RADIUS is a client/server protocol. The RADIUS client is typically a NAS, firewall, router, or VPN gateway, which requests a service such as authentication or authorization from the RADIUS server. A

RADIUS server is usually a daemon running on a UNIX machine or a service running on a Windows NT/2000 server. The daemon is software such as Cisco Secure ACS or another RADIUS server program that fulfills requests from RADIUS clients. Originally, RADIUS used UDP port 1645 for authentication traffic and 1646 for accounting traffic. However, due to an oversight in the standardization process, these ports were registered with the IANA to different services. To get around this issue, new port numbers were assigned to the RADIUS services (1812 for authentication and 1813 for accounting). However, many RADIUS implementations still use the old port numbers.

When a client needs authorization information, it passes the user credentials to the designated RADIUS server and queries it. The server then acts on the configuration information necessary for the client to deliver services to the user. A RADIUS server can also act as a proxy client to other RADIUS servers. Figure 5.2 illustrates what happens when a user attempts to log in and authenticate to a NAS using RADIUS.

Figure 5.2 Authenticating with RADIUS



The sequence of events is as follows:

1. The remote user dials into a NAS and is prompted by the NAS for credentials such as a username and password.

2. The username and encrypted password are sent from the RADIUS client (NAS) to the RADIUS server via the network.
3. The RADIUS server queries the database in which user account definitions are stored.
4. The RADIUS server evaluates the credentials and replies with one of the following responses:
   - **REJECT** The user is not authenticated; the user is prompted to re-enter the username and password. Depending on the RADIUS configuration, the user is given a certain number of tries before user access is denied.
   - **ACCEPT** The user is authenticated.
   - **CHALLENGE** A challenge is issued by the RADIUS server, with a request for additional information from the user.
   - **CHANGE PASSWORD** A request is sent from the RADIUS server specifying that the user must change his or her current password.

TACACS+

Another security protocol that is available is Terminal Access Controller Access Control System Plus (TACACS+). This should not be confused with TACACS and XTACACS, both of which are open standard protocols documented in RFC 1492 and no longer used. Despite the similar names, TACACS and XTACACS are not compatible with TACACS+. TACACS+ provides a method to validate users attempting to gain access to a service through a router or NAS. Similar to RADIUS, a centralized server running TACACS+ software responds to client requests in order to perform AAA.

NOTE

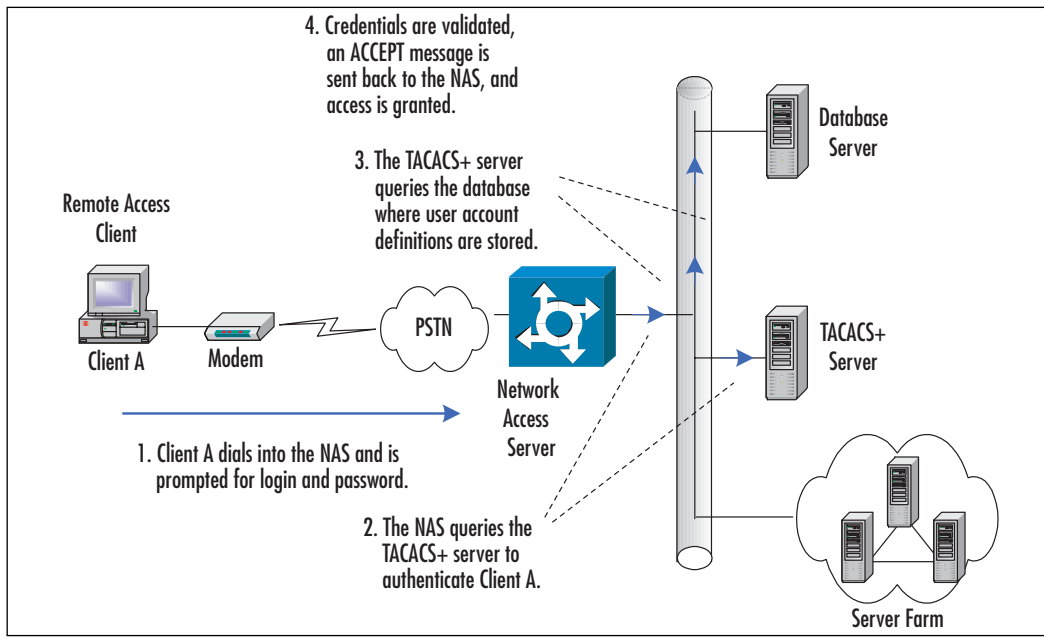
Although the specification for TACACS+ was never released as a final standards document, a draft of the specification is available at ftp://ftpeng.cisco.com/pub/tacacs/tac-rfc.1.78.txt.

TACACS+ packets rely on TCP as the transport protocol, making the connection reliable. TACACS+ can also encrypt the body of traffic travelling

between the TACACS+ server and client. Only the packet header is left unencrypted. TACACS+ allows an administrator to separate the authentication, authorization, and accounting mechanisms, thereby providing the ability to implement each service independently. Each of the AAA mechanisms can be tied into separate databases. TACACS+ uses TCP port 49 for communication.

Figure 5.3 illustrates the process that occurs when a user attempts to log in by authentication to a NAS using TACACS+:

Figure 5.3 Authenticating with TACACS+



1. When the connection is established, the NAS contacts the TACACS+ server to obtain an authentication prompt, which is then displayed to the user. The user enters his or her username, and the NAS then contacts the TACACS+ server to obtain a password prompt. The NAS displays the password prompt to the user.
2. The user enters his or her password, and these credentials are then sent to the TACACS+ daemon running on a server.
3. The TACACS+ server queries the user database and compares Client A's credentials with those stored in the database server.
4. The NAS will eventually receive one of the following responses from the TACACS+ daemon:

- **ACCEPT** The user is authenticated and the service can begin.
- **REJECT** The user failed authentication. Depending on the TACACS+ daemon, the user may be denied further access or prompted to retry the login sequence.
- **ERROR** An error occurred at some point during the authentication process. This can be either at the daemon or in the network connection between the daemon and the NAS. If an ERROR response is received, the NAS will typically try to use an alternative method for authenticating the user.
- **CONTINUE** The user is prompted for additional authentication information.

Designing & Planning…

Security Protocol Considerations

Selecting a security protocol can be a daunting task for administrators. Many factors must be taken into consideration. For example, will this security protocol facilitate only Cisco routers? Should one or two servers be dedicated in case of failure? Is one protocol easier to configure than the others?

The two most widely used security protocols are RADIUS and TACACS+. Which one should be implemented in your enterprise? Several factors will influence your decision:

- **Vendor interoperability** RADIUS enjoys support from more vendors than TACACS+.
- **Transport protocol considerations** RADIUS uses UDP as the transport layer protocol, whereas TACACS+ uses TCP, making RADIUS the faster method of the two, since UDP has less overhead. What this means is that TACACS+ traffic is more reliable than RADIUS traffic. If any disruption occurs (such as corrupted or dropped packets), TACACS+ will retransmit unacknowledged packets, whereas RADIUS will not.
- **Packet encryption** RADIUS only encrypts the password portion of the access-request packet from the AAA client to the AAA server. The rest of the packet is sent in clear text, which

Continued

can be captured and viewed by a network or protocol analyzer. TACACS+ encrypts the entire body of the packet except the TACACS+ header.

- **Overhead** RADIUS uses less CPU overhead and consumes less memory than TACACS+.
- **Authentication and authorization** RADIUS combines authentication and authorization. The access-accept packets exchanged by the RADIUS client and the server contain authorization information. This makes it difficult to separate the two elements. TACACS+ separates authentication, authorization, and accounting, allowing for advantages such as multiprotocol use. For example, TACACS+ could provide the authorization and accounting elements, and Kerberos may be used for the authorization element.
- **Protocol support** RADIUS does not support the following protocols, but TACACS+ does:
  - AppleTalk Remote Access (ARA) protocol
  - NetBIOS Frame Protocol Control protocol
  - Novell Asynchronous Services Interface (NASI)
  - X.25 PAD connection

It is also important to understand that certain features in each AAA client will only work with one of the protocols (RADIUS, or TACACS+) and not the other. For example, the PIX firewall only supports TACACS+ for authorization services and only supports RADIUS for downloadable access lists.

A detailed comparison of RADIUS and TACACS+ is available at www.cisco.com/warp/public/480/10.html.

Cisco Secure ACS for Windows

You now have a basic understanding of AAA functions and the most commonly implemented protocols (TACACS+ and RADIUS). In order to implement AAA services on the PIX firewall, you need to implement and configure an AAA server. Many AAA server products are available; the PIX firewall provides support for the following:

- Cisco Secure ACS for Windows
- Cisco Secure ACS for UNIX

- Livingston
- Merit

This chapter concentrates on Cisco Secure Access Control Server (ACS) for Windows 3.0.2 by describing its features, how to install and configure it, and how to perform basic tasks such as adding AAA clients and users.

Introduction and Features

Cisco Secure ACS for Windows is AAA server software that provides centralized user authentication, authorization, and accounting for network devices that act as AAA clients, such as routers, NASs, VPN gateways, wireless access points, and firewalls. It simultaneously supports both the TACACS+ and RADIUS protocols, allowing you to use the protocol that is most appropriate for each client. For instance, you could use TACACS+ command authorization for routers and firewalls and use RADIUS authentication for VPN access.

Cisco Secure ACS is also highly scalable, providing support for up to 500,000 users and 2000 AAA clients. An AAA server such as Cisco Secure ACS can quickly become a critical part of your infrastructure. To ensure the availability of AAA services, Cisco Secure ACS supports database replication to other ACS servers. If one server goes down, others are available to provide AAA services with current information. You can replicate all or parts of the database and can configure replication to be performed automatically at specific times (for example, every 60 minutes) or manually. For larger implementations, you can also configure a hierarchy of servers for which replication to secondary servers is initiated when a primary server completes its replication. Cisco Secure ACS provides a Web-based graphical interface, giving you the flexibility of managing the server remotely. Through the ACS interface, you can define users, groups of users, AAA clients, and external authentication databases. While Cisco Secure ACS includes its own internal user database, it also supports authentication against the following external user databases:

- Windows NT/2000 User Database
- Generic LDAP
- Novell NetWare Directory Services (NDS)
- Open Database Connectivity (ODBC) compliant relational databases
- CRYPTOCard token server
- SafeWord token server

- AXENT token server
- RSA SecureID token server
- ActivCard token server
- Vasco token server

Installing and Configuring Cisco Secure ACS

Before you install Cisco Secure ACS for Windows 3.0.2, you need to ensure that your server meets the following minimum hardware and software requirements:

- Pentium III processor, 550MHz or faster
- 256MB of RAM
- 250MB of free disk space
- Graphics resolution of 800 x 600 with 256 colors
- Windows 2000 (with SP1 or SP2), Windows 2000 Advanced Server (without Microsoft Clustering Services, and with SP1 or SP2), or Windows NT (with SP6a)
- Microsoft Internet Explorer (version 5.0 or 5.5) or Netscape Communication (version 4.76); the browser must have both Java and JavaScript enabled

NOTE

During the Cisco Secure ACS installation process, at least one AAA client (a NAS) needs to be configured on the server. If you do not have an actual NAS to configure at the time of installation, make up information just to complete the installation process. After completing installation, you can delete the "made up" NAS and create real NAS entries.

To install Cisco Secure ACS, follow these steps:

1. Log on to the server using the local administrative account and insert the Cisco Secure ACS CD into the CD-ROM drive. If the Cisco Secure ACS for Windows 2000/NT dialog box does not appear via the Windows Autorun feature, run **setup.exe** from the root directory of the

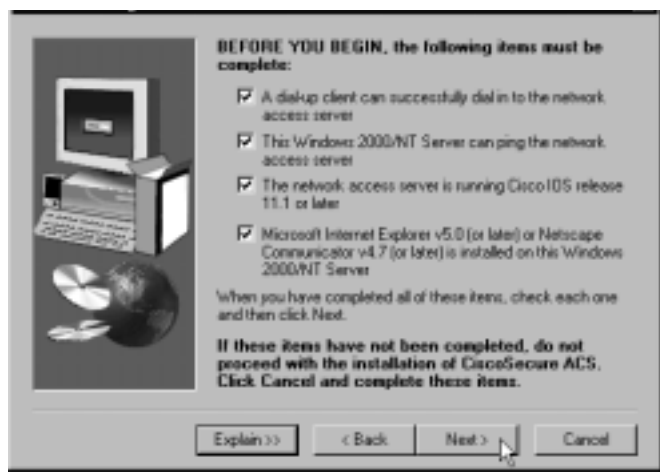
Cisco Secure ACS CD. You should now see the Cisco Secure ACS for Windows 2000/NT dialog box with the software license agreement.

2. Read the license agreement, and click the **Accept** button if you accept the terms of the license agreement. As shown in Figure 5.4, the Welcome screen should now appear.

Figure 5.4 The Cisco Secure ACS Welcome Screen

3. Click the **Next** button to display the **Before You Begin** screen (see Figure 5.5), which identifies some tasks that you must complete before installing Cisco Secure ACS.

Figure 5.5 The Cisco Secure ACS Before You Begin Screen



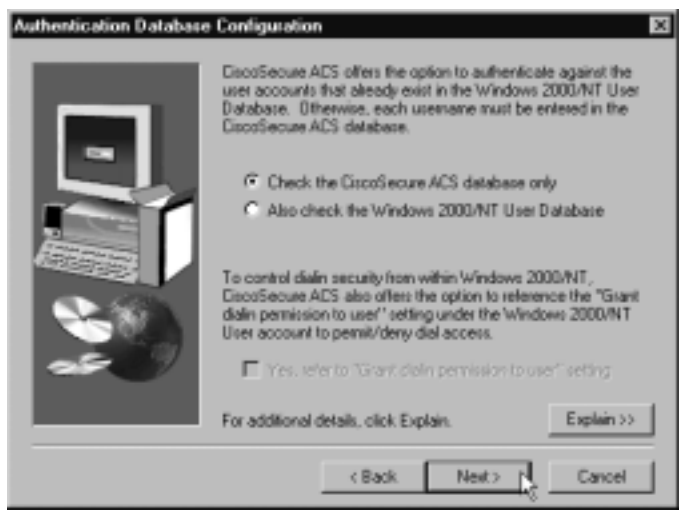
4. Review each item listed and select the corresponding check box for items that you have completed. Once all the items are checked, click the **Next** button. The Choose Destination Location screen will be displayed.

NOTE

If you have not completed *all* the items listed in the Before You Begin dialog box, click the **Cancel** button, then click **Exit Setup**. Complete the necessary items, and then restart the installation process.

5. The **Choose Destination Location** screen displays the default drive and path for the installation of Cisco Secure ACS. If you want to install the software in an alternate location, click the **Browse** button and select the desired location. Click the **Next** button to proceed to the Authentication Database Configuration screen displayed in Figure 5.6.

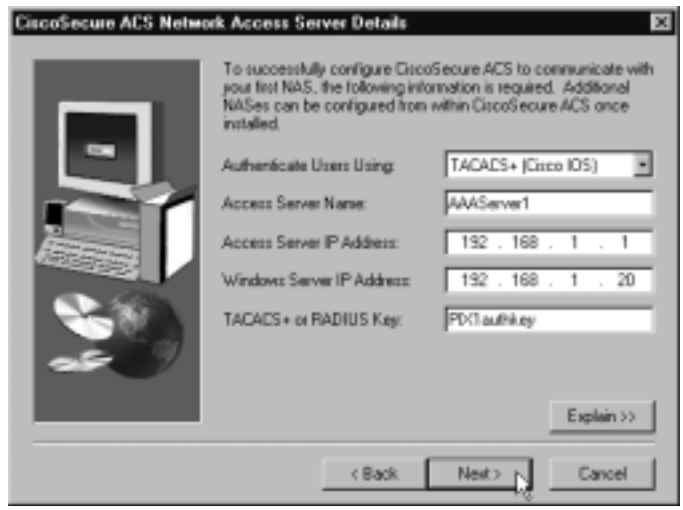
Figure 5.6 The Cisco Secure ACS Authentication Database Configuration Screen



6. The Authentication Database Configuration screen allows you to select options for authenticating users. You can choose to use the Cisco Secure ACS database only, or you can authenticate users against a Windows 2000/NT user database. Select the desired option. If you choose to include the Windows 2000/NT user database, you can then choose to

check user accounts for the "Grant dialin permission to user" setting before granting access. When this option is turned on, users will be granted access only if the "Grant dial permission to user" setting is enabled for their accounts. Otherwise, users will be denied access. Once you have selected the desired settings, click the **Next** button to proceed to the Network Access Server Details screen (see Figure 5.7).

Figure 5.7 The Cisco Secure ACS Network Access Server Details Screen



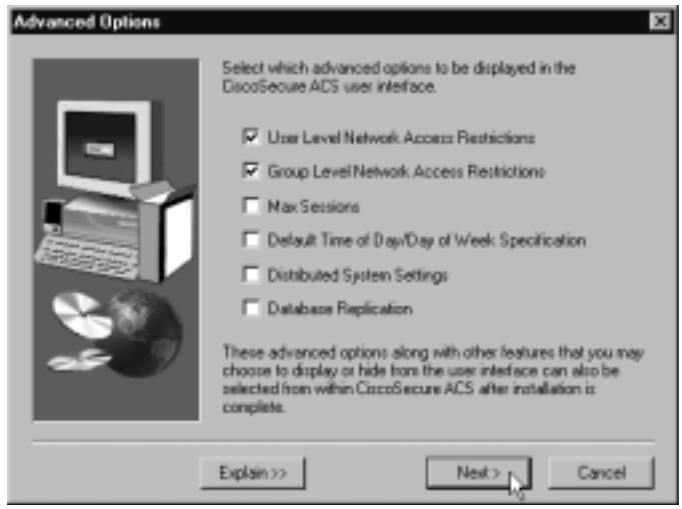
NOTE

Once you have installed Cisco Secure ACS, you can enable support for external databases, including Windows NT/2000.

7. The Network Access Server Details screen allows you to define an initial NAS (an AAA client) that will make authentication or authorization requests to the Cisco Secure ACS server. Select the appropriate authentication method in the Authenticate Users Using drop-down list. Provide the hostname of the AAA client in the Access Server Name text box. Provide the IP address of the AAA client in the Access Server IP Address text box, and provide the IP address of the server on which you are installing Cisco Secure ACS in the Windows Server IP Address text box. In the TACACS+ or RADIUS Key text box, type the key that will be

used for authentication between the AAA client and the Cisco Secure ACS server. Once you have provided the necessary AAA client details, click the Next button to proceed to the Advanced Options screen displayed in Figure 5.8.

Figure 5.8 The Cisco Secure ACS Advanced Options Screen



NOTE

The RADIUS or TACACS+ key on ACS and the AAA client must match for authentication and authorization to function correctly.

8. The Advanced Options dialog box lists several options that you can enable. These options are not enabled by default and will only appear in the Cisco Secure ACS interface if you enable them. You can always enable the desired options after installation via the Advanced Options page in the Interface Configuration section. Once you have selected the Advanced Options that you would like to enable, click the **Next** button to proceed to the Active Service Monitoring screen displayed in Figure 5.9. The Active Service Monitoring screen allows you to configure features of Cisco Secure ACS that monitor the availability of the AAA services. This screen provides you the opportunity to configure these features during the installation process, but you still have the option of configuring them any time after the installation has completed by

selecting the **System Configuration** button in the Cisco Secure ACS user interface. Click the **Explain** button for more information about the available options.

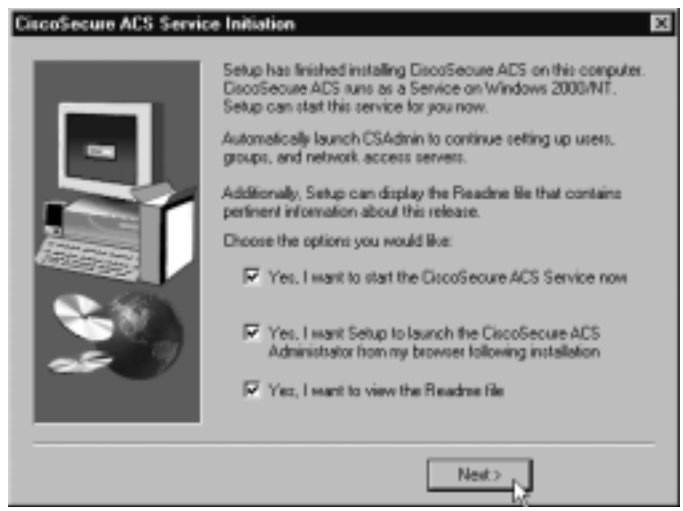
Figure 5.9 The Cisco Secure ACS Active Service Monitoring Screen

9. Once you have configured the desired service management features, click the **Next** button to proceed to the Network Access Server Configuration screen shown in Figure 5.10.

Figure 5.10 Cisco Secure ACS Network Access Server Configuration Screen

10. The Network Access Server Configuration screen appears if you selected either **TACACS+ (Cisco IOS)** or **RADIUS (Cisco IOS/PIX)** as the authentication method in the Network Access Server Details dialog box (shown in Figure 5.7). The Network Access Server Configuration screen gives you the option to configure the relevant NAS client to use the Cisco Secure ACS server AAA services. It provides you with the minimum commands necessary to enter on the Cisco device to accomplish this task and provides you an opportunity to Telnet to the device to complete the configuration. Because you have selected **TACACS+ (Cisco IOS)** as the authentication method, you will be provided with the necessary commands to configure an IOS device for TACACS+. The PIX firewall commands are different from the IOS commands, so deselect the **Yes, I want to configure Cisco IOS software now** check box. Click the **Next** button to proceed to the CiscoSecure ACS Service Initiation screen displayed in Figure 5.11.

Figure 5.11 Cisco Secure ACS Service Initiation Screen



11. The CiscoSecure ACS Service Initiation screen provides options for launching services after the installation completes. All the options are selected by default. Deselect the check boxes associated with any of the services that you do not want started. You should leave the check box associated with starting the Cisco Secure ACS service checked in order to start using Cisco Secure ACS. Once you have completed your selections, click the **Next** button to proceed to the Setup Complete screen.

12. Click the **Finish** button to complete the installation and start the service.

NOTE

To access the Cisco Secure ACS HTML interface, use the URL of http://*ip_address*:2002, where *ip_address* is the IP address of the ACS server. For example, if the ACS server has an IP address of 192.168.2.20, you would access it using the URL of http://192.168.2.20:2002.

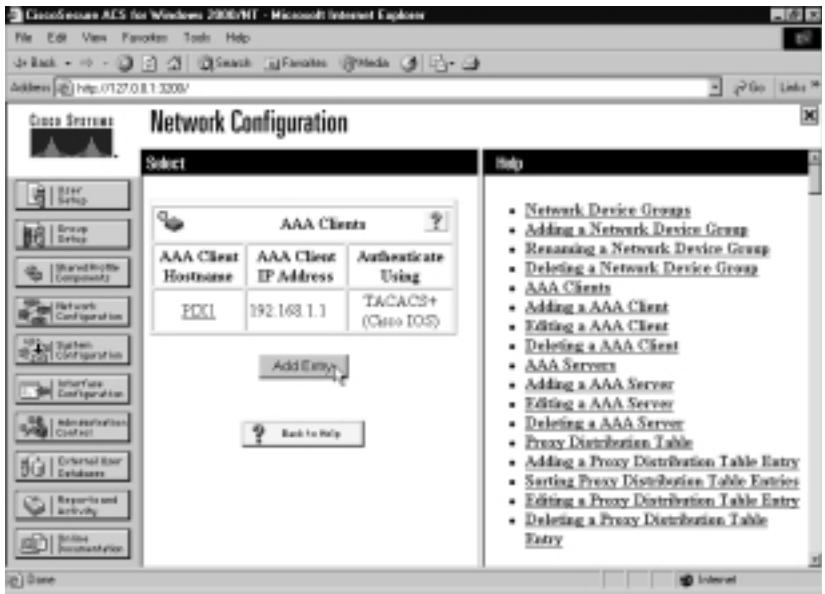
Adding an NAS to Cisco Secure ACS

To add an NAS (AAA client) to Cisco Secure ACS, you must select the **Network Configuration** button on the left side of the Cisco Secure ACS HTML interface, as shown in Figure 5.12.

Figure 5.12 Cisco Secure ACS Main Interface Network Configuration

If you are not using Network Device Groups (NDGs), a feature that allows you to manage a collection of AAA clients and servers as a single logical group, click the **Add Entry** button below the AAA clients table, as shown in Figure 5.13.

Figure 5.13 The Cisco Secure ACS Network Configuration Window without NDGs



NOTE

If you want to enable NDGs, click the **Interface Configuration** button from the main screen, click **Advanced Options**, select the **Network Device Groups** check box, and click **Submit**.

If you are using NDGs, you need to click the name of the NDG to which you want to assign the AAA client, as shown in Figure 5.14.

When the list of AAA client tables for the selected NDG appears, click the **Add Entry** button below the AAA clients table. You should now see the AAA client window shown in Figure 5.15. Enter the name and IP address of the AAA client in the **AAA Client Hostname** and **AAA Client IP Address** boxes, respectively. Enter the shared secret that the AAA client and server will use for authentication in the **Key** text box. If you have enabled NDGs, select the NDG to which this AAA client will belong from the Network Device Group drop-down list. If you have not enabled NDGs, this drop-down list will not appear on the screen. Select the authentication method that you want to use for the AAA client from the Authenticate Using drop-down list. For the PIX firewall, you will use either **TACACS+ (Cisco IOS)** or **RADIUS (Cisco IOS/PIX)**.

Figure 5.14 The Cisco Secure ACS Network Configuration Window with NDGs

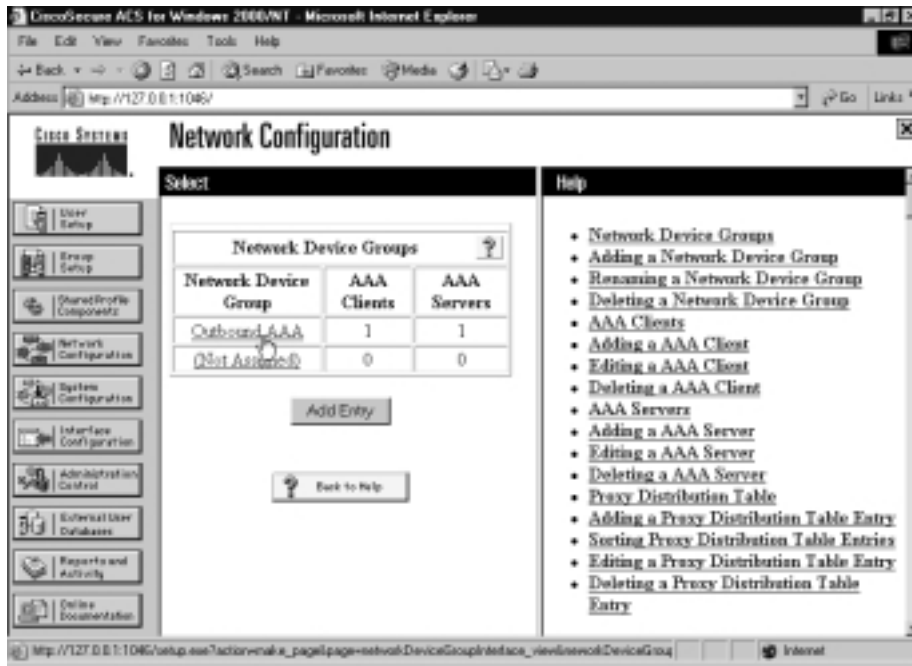
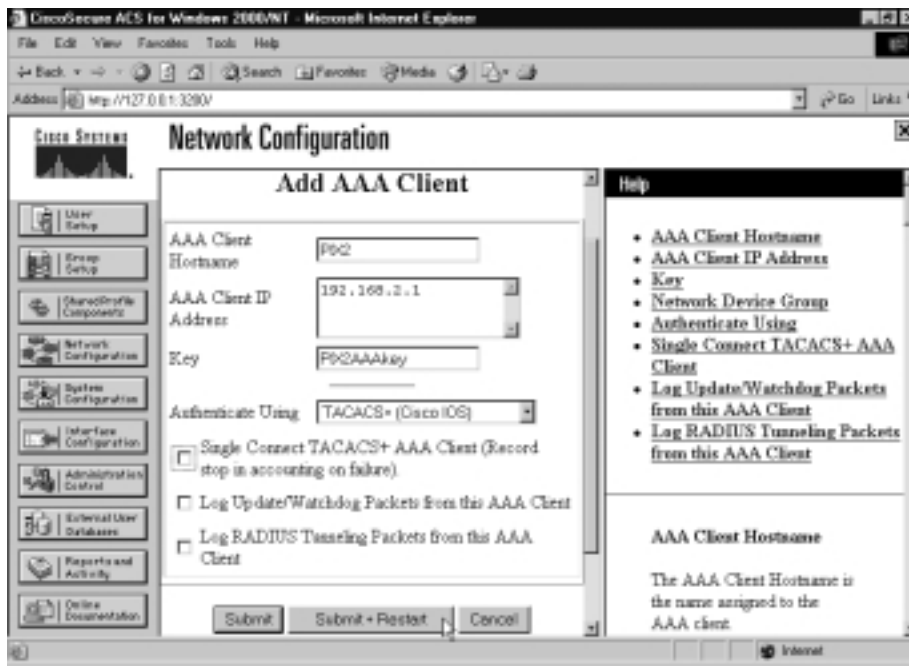


Figure 5.15 The Cisco Secure ACS AAA Client Window



After selecting the appropriate authentication method, you can choose to enable one or more of the options for communication with the AAA clients that are identified in Table 5.1.

Table 5.1 AAA Client Communication Options

Option	Description
Single Connect TACACS+ AAA Client (Record stop in accounting on failure)	Enables a single connection from the AAA client instead of a separate one for every TACACS+ request.
Log Update/Watchdog Packets from this AAA Client	Enables watchdog packets, which are sent periodically during a session and help determine the approximate length of a session when an AAA client fails and no stop packet is received.
Log RADIUS Tunneling Packets from this AAA Client	Allows RADIUS tunneling accounting packets to be logged.

After selecting any of the desired communication options, click the **Submit + Restart** button to implement the changes immediately. This choice saves the changes and restarts the Cisco Secure ACS services so that the new configuration information is loaded. If you want to save the changes but have them implemented at some point in the future, simply click the **Submit** button. In this case, when you want the changes to take effect, you must manually restart the services through the **System Configuration** | **Service Control** window.

Adding a User to Cisco Secure ACS

This section describes the basic user setup for adding a user to Cisco Secure ACS. Follow the steps described here to add a new user account. You might need to configure advanced options, depending on how the account will be used. To add a user to ACS, click the **User Setup** button on the left side of the Cisco Secure ACS HTML interface and type the desired username for the new user in the User text box, as shown in Figure 5.16. Click the **Add/Edit** button.

Within the Edit portion of the User Setup window, you can optionally enter the user's real name and a description of the account, as shown in Figure 5.17.

Figure 5.16 The Cisco Secure ACS User Setup Window

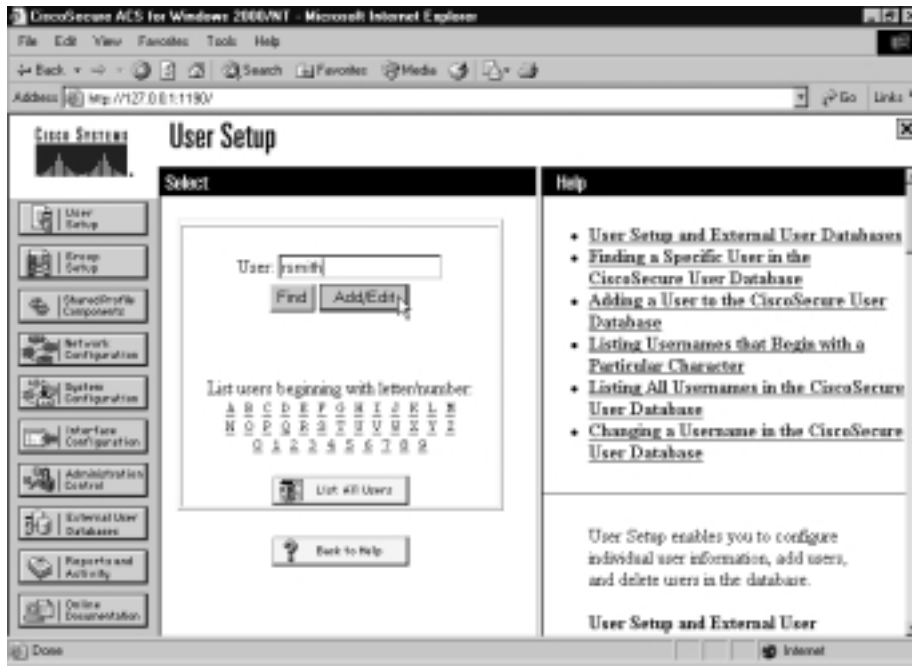
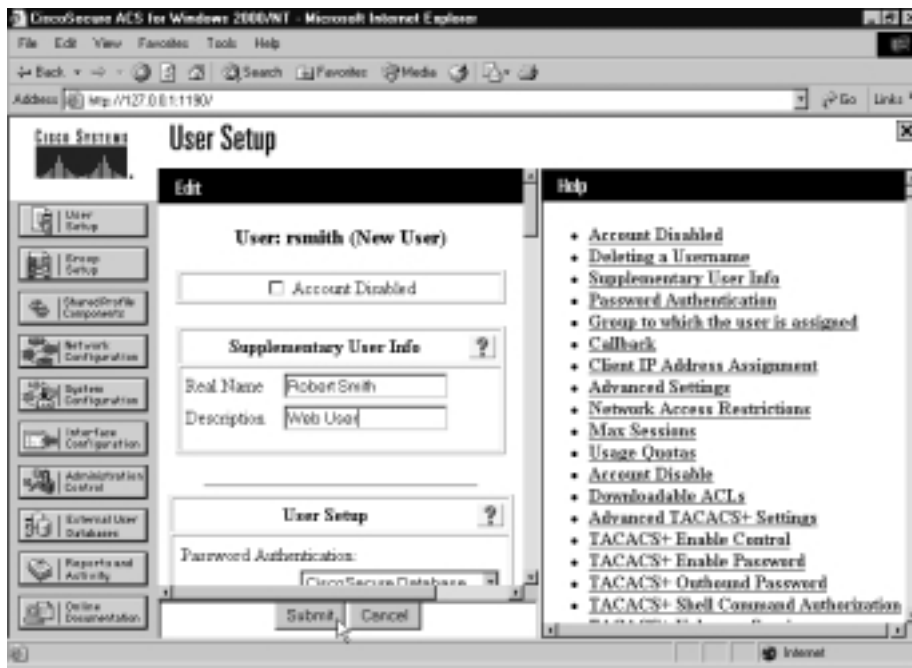
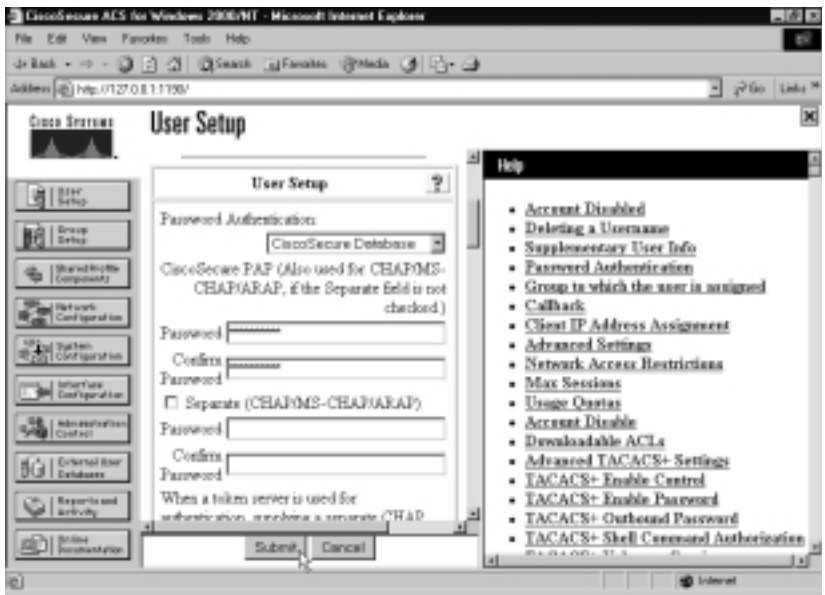


Figure 5.17 The Cisco Secure ACS Add/Edit User Window



As shown in Figure 5.18, scrolling down within the Edit portion of the User Setup window reveals additional configuration items. Select the desired authentication database from the Password Authentication drop-down list. If you're using the internal database (CiscoSecure Database), enter the user password in the **Password** field and confirm it in the **Confirm Password** field. You can configure advanced options within this window, but doing so is not necessary to establish the user account. Click the **Submit** button to complete the account definition.

Figure 5.18 The Cisco Secure ACS Add/Edit User Window



Configuring Console Authentication

As discussed previously, AAA authentication is typically used to either control access *to* a network device (for example, a PIX firewall) or to control access to network resources *through* the network device (for example, Web services through a PIX firewall). This section discusses the use of the PIX firewall AAA mechanisms to control access to the PIX firewall itself via the console port, Telnet, HTTP, or SSH. The general steps necessary to configure AAA authentication for firewall access are:

1. Configure the AAA authentication database. This database can reside locally on the firewall, or can be defined on a RADIUS or TACACS+ server.

2. Specify the methods of firewall access (serial port, Telnet, SSH, HTTP) and the AAA authentication database that should be used.

WARNING

When configuring console authentication, *do not* save your configuration until you are sure it works. If you are locked out due to a mistake, you can usually recover access by simply restarting the PIX firewall from the configuration that is saved in flash memory.

Configuring Local Console Authentication

If you are configuring the PIX firewall to use a local database to authenticate users attempting to access the firewall itself, you should use the following command to define users on the firewall:

```
username <username> {nopassword | password <password> [encrypted]}
    [privilege <level>]
```

Specify the username that you want to assign to the user. Use the *nopassword* keyword to create a local account with no password. Use the *password* keyword to assign a password to a local account, and specify the password. If the password that you are specifying is already encrypted, use the *encrypted* keyword. To assign a privilege level to the user account, use the *privilege* keyword and specify the desired level between 0 and 15. Privilege levels are discussed in detail later in this chapter. To delete a user, use the following command:

```
no username <username>
```

To view a list of configured usernames, use the following command:

```
show username [<username>]
```

To remove the entire user database, use the *clear username* command in Configuration mode.

Once you have defined the local users, you need to specify that the local database should be used for the various access methods by executing the following command:

```
aaa authentication [serial | enable | telnet | ssh | http] console LOCAL
```

NOTE

The term *console* here does not mean the console port on the PIX firewall. It refers to any administrative session to the PIX firewall, such as SSH or HTTP.

Use the *serial*, *enable*, *telnet*, *ssh*, or *http* keywords to specify the access method that requires authentication. For example, you can issue the following commands to establish a local user account and specify that the local database should be used when a user attempts to access the PIX firewall via Telnet, SSH, or HTTP (PDM):

```
PIX1(config)# username pixadm password pixpassword
PIX1(config)# aaa authentication telnet console LOCAL
PIX1(config)# aaa authentication ssh console LOCAL
PIX1(config)# aaa authentication http console LOCAL
```

The enable and SSH access methods allow three tries before denying authentication. Serial and Telnet continue to prompt the user until a successful login takes place.

Configuring RADIUS and TACACS+ Console Authentication

If you are configuring the PIX firewall to use RADIUS or TACACS+ to authenticate users attempting to access the firewall itself, first use the following command to define a *group* for the AAA servers that the firewall will use:

```
aaa-server <group_tag> protocol <auth_protocol>
```

Specify a name for the server group (**group_tag**) and either **tacacs+** or **radius** as the authentication protocol (**auth_protocol**).

NOTE

You can specify up to 14 AAA servers groups on a PIX firewall. The *clear aaa-server* command is used to remove an AAA server group.

Then use the following command to define specific AAA servers that will be associated with the group:

```
aaa-server <group_tag> [(interface)] host <server_ip> [<key>] [timeout
    <seconds>]
```

Specify the name of the group (**group_tag**) to which the server will belong and the name of the interface (**interface**) on which the server will reside. If the **interface** is not specified, it is assumed to be the inside interface. Use the *host* keyword to specify the IP address of the AAA server. Specify the secret key that will be used between the AAA client and the server. If the key is not specified, the PIX will use Unencrypted mode to communicate with the AAA server. Use the *timeout* keyword to specify the duration that the PIX firewall waits to retry access. The PIX will retry four times before choosing the next server to attempt authentication. The default value for the timeout is 5 seconds, and the maximum allowed is 30 seconds. You can specify a maximum of 16 AAA servers in a group. To remove a server from the configuration, use the *no aaa-server* command.

NOTE

By default, the PIX firewall communicates to RADIUS servers on port 1645 for authentication and port 1646 for accounting. Newer RADIUS servers may use port numbers 1812 and 1813. If your server uses ports other than 1645 and 1646, you should define ports appropriately on the PIX firewall using the *aaa-server radius-authport* and *aaa-server radius-acctport* commands before defining the RADIUS servers with the *aaa-server* command.

Once you have designated AAA authentication servers using the *aaa-server* command, you can verify your configuration using the *show aaa-server* command. The next step is to specify the AAA authentication database that should be used for the various access methods. Use the following command to specify the authentication database:

```
aaa authentication [serial | enable | telnet | ssh | http] console
    <group_tag>
```

The syntax is very similar to using local authentication. The *group_tag* parameter identifies the AAA server group to use for authentication. For example, you can issue the following commands to create the AuthPIX server group, assign a TACACS+ server to it, and specify that the group should be used when a user attempts to access the PIX firewall via Telnet, SSH, and HTTP:

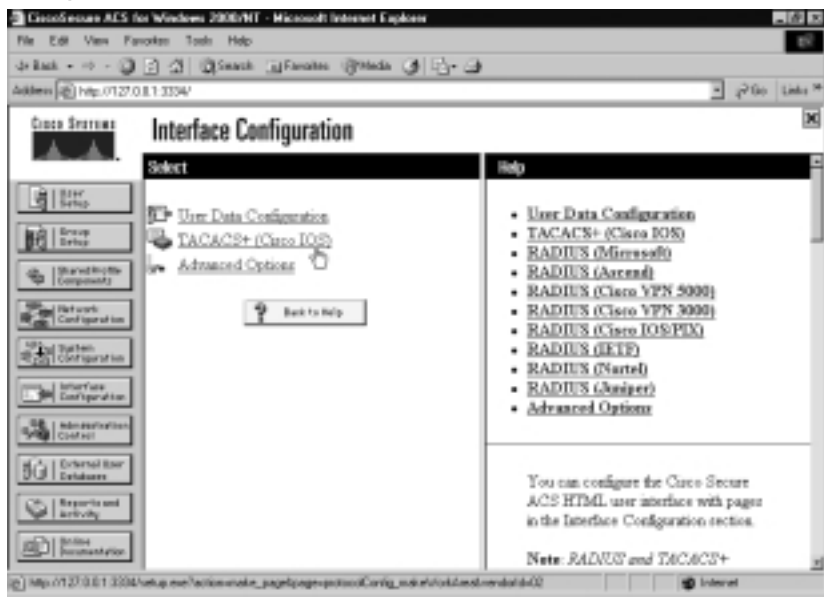
```
PIX1(config)# aaa-server AuthPIX protocol tacacs+
PIX1(config)# aaa-server AuthPIX (inside) host 10.5.1.20 TacacsKey
PIX1(config)# aaa authentication telnet console AuthPIX
PIX1(config)# aaa authentication ssh console AuthPIX
PIX1(config)# aaa authentication http console AuthPIX
```

Configuring TACACS+ Enable Console Authentication in Cisco Secure ACS

To configure enable console authentication using TACACS+ in Cisco Secure ACS, you must enable Advanced TACACS+ Features, configure the enable privileges for the desired users or groups, and then configure enable authentication on the PIX Firewall. The following paragraphs describe the steps you need to take to complete the configuration.

To enable Advanced TACACS+ Features, click **TACACS+ (Cisco IOS)** within the Interface Configuration window, as shown in Figure 5.19.

Figure 5.19 Cisco Secure ACS Interface Configuration Window TACACS+ (Cisco IOS) Options



Within the TACACS+ (Cisco IOS) options window, scroll down to the Advanced Configuration Options section and select the **Advanced TACACS+**

Features check box, as shown in Figure 5.20. Click the **Submit** button to enable the advanced features.

Figure 5.20 Cisco Secure ACS: Enabling Advanced TACACS+ Features

To provide a selected user the ability to enter the PIX firewall Privileged mode, navigate to the user's profile via the User Setup window and scroll down to the Advanced TACACS+ Setting section of the window. Under the TACACS+ Enable Control subsection, you have four options for specifying the user's maximum possible privilege. These options are identified and described in Table 5.2.

Table 5.2 TACACS+ Enable Control Options

TACACS+ Enable Control Option	Description
Use Group Level Setting	Determines the user's maximum privilege level based on the corre sponding group settings.
No Enable Privilege	Provides the user with no enable privileges. This is the default option.
Max Privilege for any AAA Client	Specifies the maximum privilege for the user when accessing any AAA client device.

Continued

Table 5.2 Continued

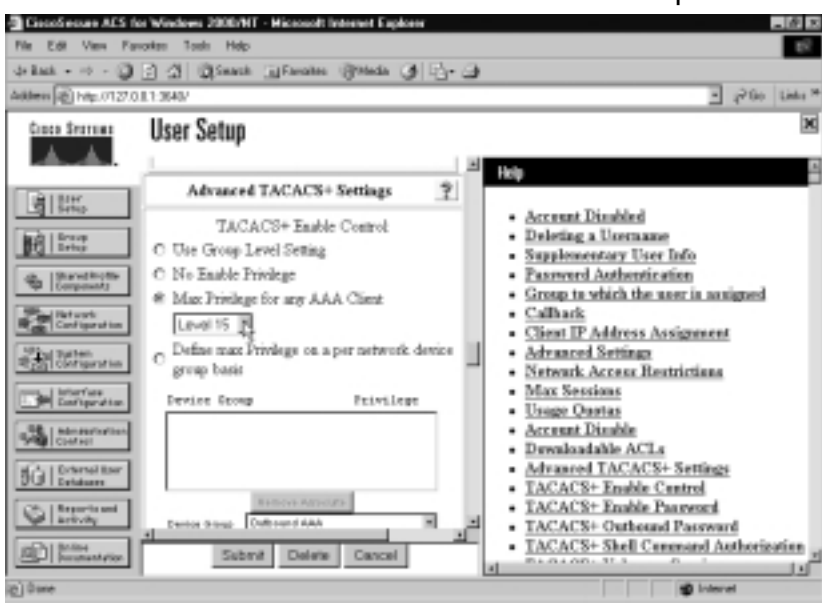
TACACS+ Enable Control Option	Description
Define Max Privilege on a per network device group basis	Specifies the maximum privilege for the user based on NDGs. Note that NDGs must be enabled in order to use this option. See the section titled "Adding an NAS to Cisco Secure ACS" for infor mation on how to enable NDGs.

NOTE

The privilege specifies the level of access available to the user and is discussed later in this chapter.

Select the **Max Privilege for any AAA Client** radio button and choose **Level 15** from the corresponding drop-down list, as shown in Figure 5.21.

Figure 5.21 Cisco Secure ACS TACACS+ Enable Control Options



Scroll further down to the TACACS+ Enable Password section of the window (see Figure 5.22), and select the desired password scheme for entering Privileged mode. Table 5.3 identifies and describes the TACACS+ enable password options.

Figure 5.22 Cisco Secure ACS TACACS+ Enable Password Options

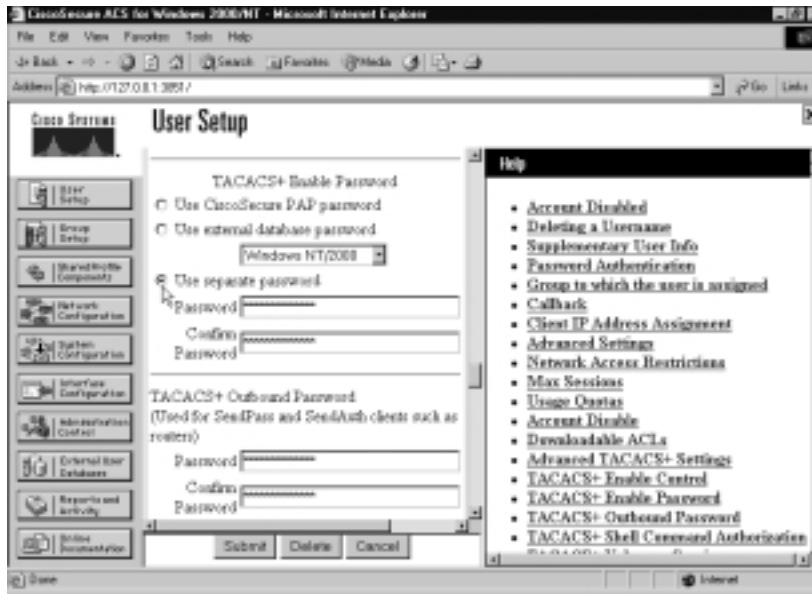


Table 5.3 TACACS+ Enable Password Options

TACACS+ Enable Password Option	Description
Use CiscoSecure PAP password	Use the Cisco Secure password defined during the basic user setup, as described in the section titled "Adding a User to Cisco Secure ACS."
Use external database password	Use an external database as the source of the enable password, and select the appropriate database from the corre sponding drop-down list.
Use separate password	Specify a separate password by typing and retyping the password in the corre sponding text boxes.

Click the **Submit** button to complete the Cisco Secure ACS configuration. To configure TACACS+ enable authentication on the PIX Firewall, use the *aaa*

authentication enable console command, as described previously. For example, to configure TACACS+ enable authentication using a previously defined TACACS+ server group called *TACACSGroup*, issue the following command on the PIX firewall:

```
PIX1(config)# aaa authentication enable console TACACSGroup
```

Configuring Command Authorization

As discussed previously, AAA authorization is typically used to authorize either user actions attempted while logged into a network device (such as a PIX firewall) or attempts to use network services. This section discusses the use of the PIX firewall AAA mechanisms to control user actions on the firewall itself, sometimes called *command authorization*.

Beginning with version 6.2, the PIX introduces support for up to 16 privilege levels so that you can define and assign users privileges based on what is necessary to accomplish their duties. This is similar to what has been available with Cisco IOS software. Sixteen privilege levels (0 through 15) are available, and the higher the privilege level, the more access the level has. By default, most PIX firewall commands are assigned to Privilege Level 15 (commonly referred to as Enable or Privileged mode), with only a few assigned to Privilege Level 0. No commands are assigned to privilege levels between 1 through 14. You do not have to give a user full privileged access to the PIX firewall if the user only needs to execute a small subset of commands. This is normally accomplished by moving commands from Privilege Level 15 into lower privilege levels. You also have the option of moving commands from Privilege Level 0 into higher privilege levels. You can implement command authorization using either the PIX firewall local database or an AAA server.

Regardless of the method you choose, the general steps you need to follow in configuring AAA authorization on the PIX firewall are as follows:

1. Assign commands to appropriate privilege levels. If you are enabling authorization using the PIX firewall local database, use the *privilege* command. If you are enabling AAA authorization using an AAA server, use the appropriate mechanism provided by the server.
2. Define user accounts assigned to appropriate privilege levels. If you are enabling AAA authorization using the PIX firewall local database, use the *username* command. If you are enabling AAA authorization using an AAA server, use the appropriate mechanism provided by the server.

3. Enable AAA authorization on the PIX firewall. Regardless of whether you are enabling AAA authorization using the PIX firewall local database or an AAA server, use the *aaa authorization* command.

WARNING

When configuring command authorization, *do not* save your configuration until you are sure it works. If you are locked out due to a mistake, you can usually recover access by simply restarting the PIX firewall from the configuration that is saved in flash memory.

Configuring Local Command Authorization

To implement command authorization using the PIX firewall local database, you must first assign the various commands to appropriate privilege levels using the following command:

```
privilege [show | clear | configure] level <level> [mode {enable |
    configure}] command <command>
```

Pick the appropriate command for which to set a privilege level (*show, clear,* or *configure*, or blank if it is not one of these). The *level* parameter specifies the privilege level to which to assign the command. The *mode* parameter specifies the mode (*enable* or *configure*) to which the specified level applies. Finally, *command* is the command you are adding to the privilege level.

Once you have assigned commands to the desired privilege levels, you need to assign users to the appropriate privilege levels based on those users' duties. If you are using the local database, use the *username* command with the *privilege* keyword. The *username* command syntax was described previously in this chapter.

Now that you have assigned both commands and users to appropriate privilege levels, you are ready to enable AAA authorization on the PIX firewall using the following command:

```
aaa authorization command LOCAL
```

Here is an example:

```
PIX1(config)# privilege show level 10 command access-list
PIX1(config)# privilege configure level 11 command access-list
```

```
PIX1(config)# privilege clear level 12 command access-list
PIX1(config)# username dora password wedidit privilege 12
PIX1(config)# username bootes password abre privilege 11
PIX1(config)# username swiper password noswiping privilege 10
PIX1(config)# aaa authorization command LOCAL
```

The *privilege* commands assign different command modifiers of the access-list command to different privilege levels. The *username* commands defines users and assigns them privilege levels. Finally, the *aaa authorization command* command enables *local* user authorization services. The result is that the user dora is authorized to configure, clear, and show access lists, the user bootes is authorized to configure and show access lists, and the user swiper is authorized only to show access lists.

To determine the privilege level to which a particular command is assigned, use the following command:

```
show privilege command <command>
```

To determine the commands assigned to a particular level, use the following command:

```
show privilege level <level>
```

To show all the commands and the levels to which they are assigned, use the following command:

```
show privilege all
```

Configuring TACACS+ Command Authorization

In addition to local AAA authorization, the PIX firewall can be configured to use TACACS+ AAA authorization. The primary advantage of using TACACS+ authorization rather than local authorization is that you can leverage the central TACACS+ database for multiple PIX firewalls without having to recreate a potentially complex configuration on each firewall.

NOTE

The PIX firewall does not support RADIUS for command authorization.

When you implement TACACS+ command authorization on the PIX firewall, it sends the username, command, and command modifier (for example, *show, clear, no*) to the TACACS+ server for authorization. This occurs for each command that a user enters on the PIX firewall. Note that the information sent to the TACACS+ server does not include all the arguments that the user entered as part of the command.

To implement command authorization using a TACACS+ server, you must perform the following tasks:

1. Configure enable console authentication using TACACS+, as described in the section titled "Configuring TACACS+ Enable Console Authentication."
2. Configure Cisco Secure ACS to support TACACS+ command authorization.
3. Define the desired shell command authorization set.
4. Assign the shell command authorization set to the desired users or groups.
5. Enable command authorization on the PIX firewall using the *aaa authorization command* command.

Configuring Cisco Secure ACS to Support TACACS+ Command Authorization

Before configuring command authorization for groups or users, you need to enable per-user TACACS+/RADIUS attributes by navigating to the **Interface Configuration | Advanced Options** window and selecting the **Per-user TACACS+/RADIUS Attributes** check box, as shown in Figure 5.23. Click the **Submit** button to complete the configuration.

You also need to enable the TACACS+ shell (exec) option by navigating to the **Interface Configuration | TACACS+ (Cisco IOS)** window and selecting the **User** and/or **Group** check boxes next to the Shell (exec) option, as shown in Figure 5.24. Click the **Submit** button to complete the configuration.

Figure 5.23 Cisco Secure ACS: Enabling Per-User TACACS+/RADIUS Attributes

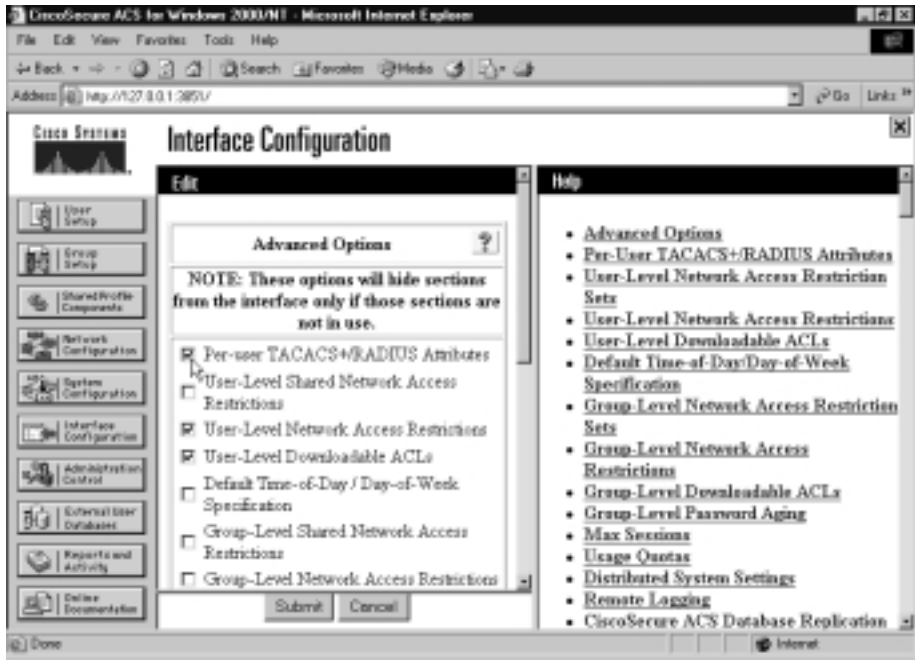
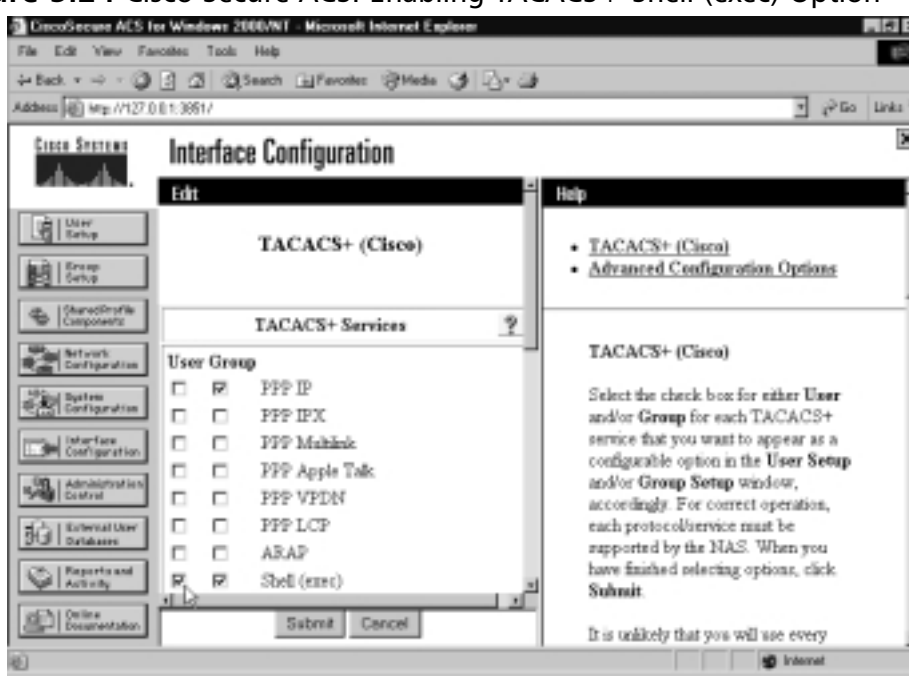


Figure 5.24 Cisco Secure ACS: Enabling TACACS+ Shell (exec) Option



Defining the Shell Command Authorization Set

Now that you have enabled the necessary options within the Cisco Secure ACS HTML interface, you are ready to define the shell command authorization set that identifies the commands a user can use. Navigate to the **Shared Profile Components** window and select **Shell Command Authorization Sets**, as shown in Figure 5.25.

Figure 5.25 Cisco Secure ACS: Shared Profile Components Window

Within the Shell Command Authorization Sets window, click the **Add** button to define a new authorization set, as shown in Figure 5.26.

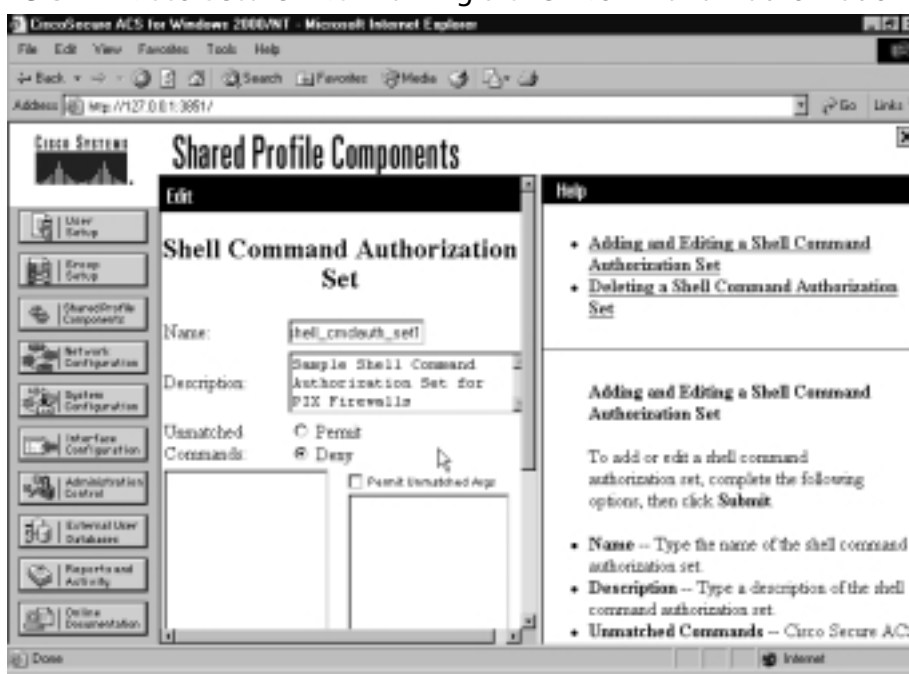
Within the Shell Command Authorization Set Edit window, type the name of the command set in the **Name** text box and an optional description in the **Description** text box, as shown in Figure 5.27.

NOTE

The name of the shell command authorization set can contain up to 32 characters without any leading or trailing spaces. The following special characters cannot be used in the name: # ? " * > <

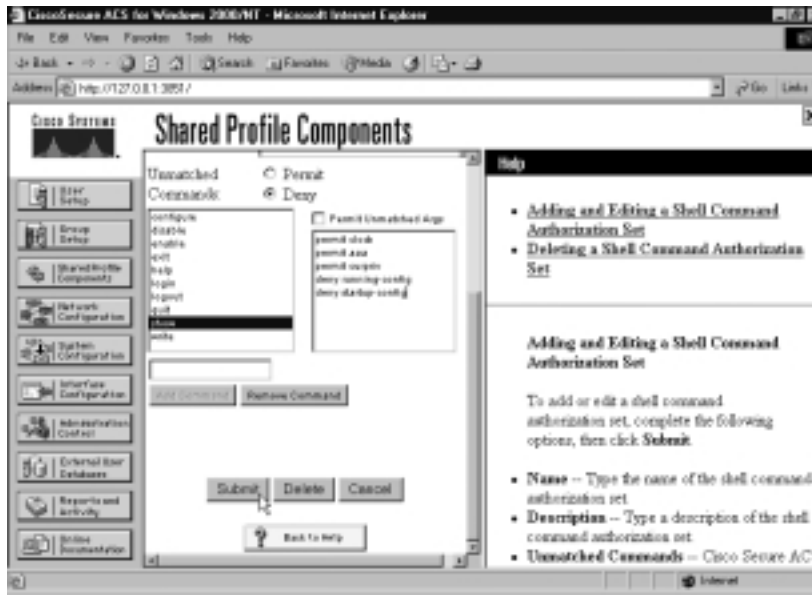
Figure 5.26 Cisco Secure ACS: Shell Command Authorization Sets Window

Figure 5.27 Cisco Secure ACS: Naming a Shell Command Authorization Set



Scroll down within the Shell Command Authorization Set Edit window and define the command authorization set, as shown in Figure 5.28. The command authorization set is a list of commands and arguments that a user is allowed to execute. You begin creating the list by selecting the action you want taken for any attempted commands that do not match one on the command authorization set. Select either **Permit** or **Deny** from the **Unmatched Commands** radio button, as shown in Figure 5.28. You can now start building the list by typing a command in the text box and clicking the **Add Command** button. Do this for each command that you want listed in the authorization set.

Figure 5.28 Cisco Secure ACS: Defining a Shell Command Authorization Set



NOTE

When adding a command to the list, make sure that you enter the command only (with no arguments). You will have a chance to permit or deny both specific arguments and unmatched arguments within the command.

For each command, you can list specific arguments that you want to permit or deny by highlighting the command and in entries in the list box to the right of the command, as shown in Figure 5.28. Each entry should have the following format:

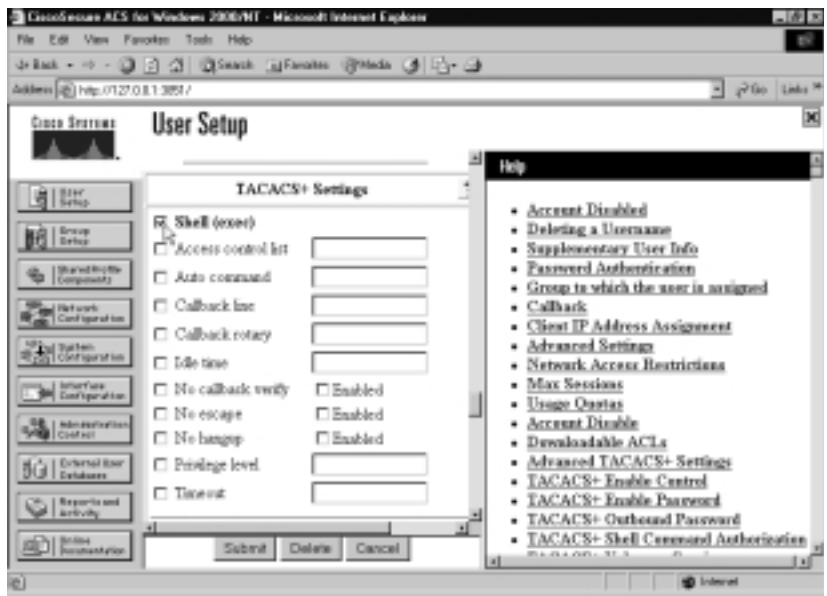
```
{permit | deny} <argument>
```

In addition, if you do not want to exhaustively list each possible command argument that you might want to allow, you can simply check the **Permit Unmatched Args** check box to the right of the highlighted command. When you are finished defining your command authorization set, click the **Submit** button to complete the configuration.

Assigning the Command Authorization Set to Users or Groups

Now that you have defined the shell command authorization set, you can assign it to users and/or groups. Navigate to the desired user or group via the User Setup or Group Setup window. Scroll down within the window to the TACACS+ Settings section of the window. Check the **Shell (exec)** check box, as shown in Figure 5.29.

Figure 5.29 Cisco Secure ACS: Assigning Command Authorization Sets



To continue with the assignment of a command authorization set, scroll further down within the TACACS+ Settings section of the window to the Shell Command Authorization Set area, as shown in Figure 5.30. Table 5.4 identifies and describes the four options for assigning a command authorization set.

Table 5.4 Cisco Secure ACS: Command Authorization Set Assignment Options

Command Authorization Set Assignment Option	Description
None	Assigns no command authorization set. This is the default option.
As Group	Determines the user's command authorization set based on the corresponding group settings.
Assign a Shell Command Authorization Set for any network device	Specifies the command authorization set to apply to the user regardless of the AAA client device that the user is accessing.
Assign a Shell Command Authorization Set on a per Network Device Group basis	Specifies the command authorization set to apply to the user based on NDGs. Note that NDGs must be enabled in order to use this option. See the section entitled "Adding a NAS to Cisco Secure ACS" for information on how to enable NDGs.

Figure 5.30 illustrates the selection of the **Assign a Shell Command Authorization Set for any network device** option and the selection of a command authorization set from the corresponding drop-down list. Click the **Submit** button to complete the configuration.

Figure 5.30 Cisco Secure ACS: Assigning Command Authorization Sets

Enabling Command Authorization on the PIX Firewall

To complete the configuration of TACACS+ command authorization, you need to configure the PIX firewall to check with the TACACS+ server to determine if a user is authorized to execute particular commands. To accomplish this task, use the following command:

```
aaa authorization command <tacacs_server_tag>
```

The *tacacs_server_tag* parameter specifies the name of the TACACS+ server group. For example, to complete the configuration of TACACS+ command authorization using a previously defined TACACS+ server group called *TACACSGroup*, issue the following command on the PIX firewall:

```
PIX1(config)# aaa authorization command TACACSGroup
```

Configuring Authentication for Traffic Through the Firewall

The PIX firewall can provide authentication and authorization of user attempts to access services *through* the PIX firewall. Specifically, the PIX firewall allows you to implement authentication and authorization for inbound or outbound HTTP, FTP, and Telnet sessions. This functionality is provided through the cut-through proxy functionality. In addition, the PIX can provide support for other types of services using virtual Telnet.

Configuring Cut-Through Proxy

Cut-through proxy allows you to control services available through the firewall by user rather than by IP address, providing a finer granularity of control. User connection requests can be authenticated or authorized against either a TACACS+ or a RADIUS server. One of the most impressive features of cut-through proxy is its performance. In traditional proxy-based firewalls, every data packet in a session needs to be processed at the application layer, resulting in tremendous overhead and low performance. Using cut-through proxy functionality, the PIX transparently authenticates and authorizes the initial connection attempt at the application layer. Once authentication and/or authorization have been performed, the session is shifted and traffic flows directly between the two hosts while state information is maintained, providing a significant performance advantage over proxy firewalls.

NOTE

You cannot use the local database for authentication of traffic *through* the PIX firewall.

To implement AAA authentication to control user access to services *through* the PIX firewall, you need to complete the following high-level tasks:

1. Define the PIX firewall appropriately as an AAA client to your AAA server. See the section titled "Adding a NAS to Cisco Secure ACS" for a description of how to accomplish this task if you are using Cisco Secure ACS as your AAA server. Make sure that you define the appropriate authentication method (for example, TACACS+ or RADIUS) when you define the PIX as an AAA client on your Cisco Secure ACS server.
2. Define the users appropriately within the AAA server. See the section entitled "Adding a User to Cisco Secure ACS" for a description of how to accomplish this task if you are using Cisco Secure ACS.
3. Define the AAA server group and AAA servers on the PIX firewall using the *aaa-server* command, as discussed previously.
4. Enable and configure AAA authentication on the PIX firewall using the *aaa authentication* command syntax to control user access to services *through* the PIX firewall. The syntax of this command is as follows:

```
aaa authentication {include | exclude} <authen_service> {inbound |
    outbound | <interface>} <local_ip> <local_mask> <foreign_ip>
        <foreign_mask> <group_tag>
```

Use the *include* keyword to create a new rule and the *exclude* keyword to create an exception to a previous rule. The *authen_service* parameter needs to be *any, ftp, http,* or *telnet*. The *inbound* or *outbound* keywords specify inbound or outbound services, respectively. The *interface* parameter specifies the interface from which to authenticate connections. The *local_ip* and *local_mask* parameters specify the host or network that you want authenticated. To specify all hosts, use 0 for both. The *foreign_ip* and *foreign_mask* parameters specify the host or network that you want to access *local_ip.* To specify all hosts, use 0 for both. Lastly, *group_tag* specifies the AAA server group to use for authentication.

Configuring & Implementing…

Setting Authorization Timers

Although it is not necessary to configure and implement cut-through proxy authentication, the *uauth* timer is an important feature to understand to ensure that your proxy authentication functions in the intended manner. The *uauth* timer controls how frequently users need to reauthenticate. When a user is authenticated via the cut-through proxy, the PIX firewall caches successful authentication for a time period determined by this timer. Once the time period expires, the user is required to reauthenticate by providing the username and password information again. The PIX firewall does not prompt the user for the authentication information immediately after the *uauth* timer expires. It prompts the user for the authentication information only when a connection is attempted after the timer expires.

The *uauth* timer has two qualifiers that you can configure separately to control reauthentication: *inactivity* and *absolute*. The *inactivity* qualifier requires users to reauthenticate after a specified period of inactivity; the *absolute* qualifier requires users to reauthenticate after an absolute period of time. The following general guidelines should be followed regarding the configuration of the *uauth* inactivity and absolute timers:

- Setting both timers to 0 disables authentication caching and requires the user to authenticate for every connection attempt.
- Do not set both timers to 0 if passive FTP is being used through the PIX firewall.
- Do not set both timers to 0 if the *virtual* command is used for Web authentication. (See the section titled "Virtual HTTP" for details.)
- To reauthenticate users only after a period of inactivity, set the inactivity timer to the desired duration and set the absolute timer to 0.
- Both timers can be configured, but ensure that the duration for the absolute timer is greater than the duration for the inactivity timer; otherwise, the inactivity timer will never be used.

Continued

The syntax for setting the *uauth* timers is:

```
timeout uauth <hh:mm:ss> [absolute | inactivity]
```

If the *absolute* or *inactivity* keywords are not used, the *absolute* timer is adjusted. To view the timeout values, use the following command:

```
show timeout uauth
```

Security Alert

It is possible for someone to launch a denial of service (DoS) attack on the PIX firewall by initiating many login attempts on the AAA authentication mechanisms without providing any login information. Each login attempt creates a connection that will remain open until a PIX timeout expires. By initiating enough attempts, the attacker could exhaust AAA resources so that no further login attempts can be serviced. The PIX firewall has a feature called Floodguard, which protects against this attack by reclaiming resources that are not in an active state. Floodguard is enabled by default. You can find more details on this feature in Chapter 4.

Here is an example of AAA authentication for Telnet services through the firewall:

```
PIX1(config)# aaa-server AuthOut protocol tacacs+
PIX1(config)# aaa-server AuthOut (inside) host 192.168.1.20 PIX1authkey
PIX1(config)# aaa authentication include telnet outbound 0 0 0 0 AuthOut
```

In this example, cut-through proxy is enabled for Telnet from any host to any host. After completing the configuration, any outbound Telnet session to a device through the PIX firewall results in an authentication challenge from the PIX firewall, and then the user will be connected to the device to which the user initiated the session. For example, Figure 5.31 shows a successful Telnet connection through the PIX firewall to a Cisco router. The user authenticates against the PIX firewall, and then the Telnet session to the router is established.

Figure 5.31 Cut-Through Proxy Telnet Prompt

Here's an example of AAA authentication for FTP services through the firewall:

```
PIX1(config)# aaa-server AuthOut protocol tacacs+
PIX1(config)# aaa-server AuthOut (inside) host 192.168.1.20 PIX1authkey
PIX1(config)# aaa authentication include ftp outbound 0 0 0 0 AuthOut
```

In this example, any outbound FTP session to a host through the PIX firewall results first in an authentication challenge from the PIX firewall, then an authentication challenge from the device to which the user is connecting. For example, Figure 5.32 shows the cut-through proxy authentication prompt for an FTP connection request through the PIX firewall.

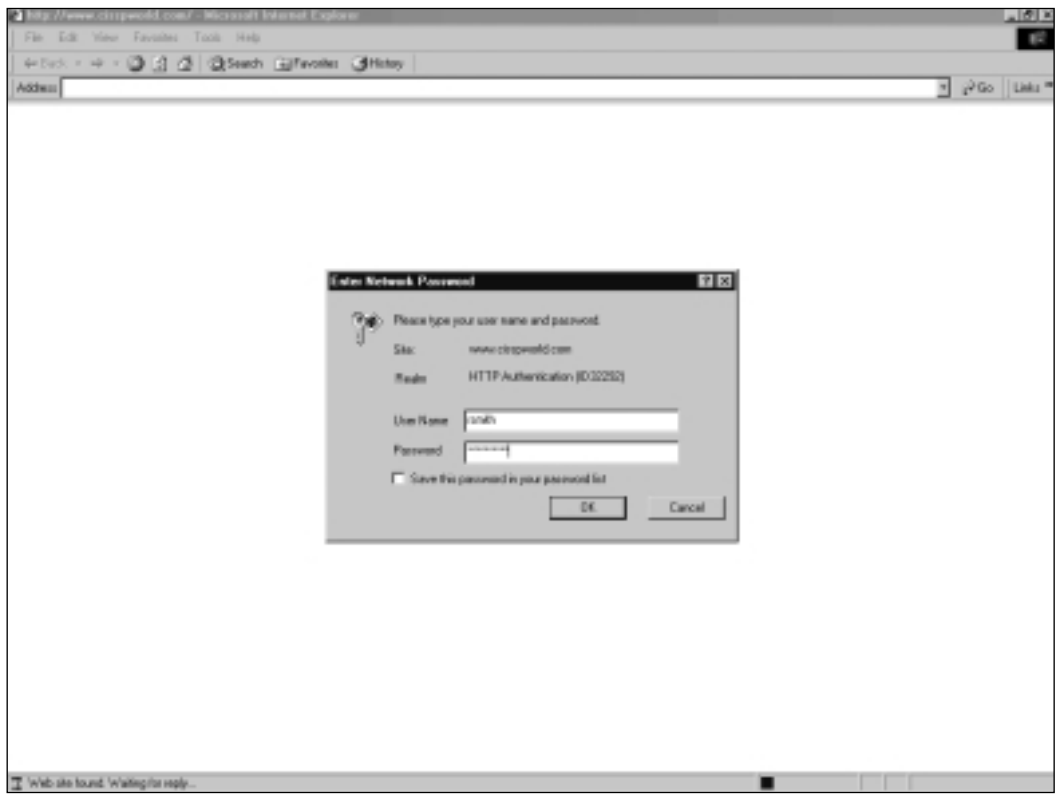
Figure 5.32 Cut-Through Proxy FTP Prompt

Here is an example of AAA authentication for HTTP services through the firewall:

```
PIX1(config)# aaa-server AuthOut protocol tacacs+
PIX1(config)# aaa-server AuthOut (inside) host 192.168.1.20 PIX1authkey
PIX1(config)# aaa authentication include http outbound 0 0 0 0 AuthOut
```

In this example, any outbound HTTP session to a host through the PIX firewall results first in an authentication challenge from the PIX firewall, then a session is established to the device to which the user is connecting. The HTTP host the user is connecting to may reprompt for authentication. For example, Figure 5.33 shows the authentication prompt for an HTTP connection request through the PIX firewall.

Figure 5.33 Cut-Through Proxy HTTP Prompt



Virtual HTTP

With cut-through proxy authentication enabled for Web traffic (HTTP), users may experience some problems when connecting to Web sites that run Microsoft IIS with Basic Authentication or NT Challenge enabled. This is an issue when the Web server requires different login credentials from the PIX firewall's AAA server. When using HTTP authentication on a Microsoft IIS Web site with Basic Authentication or NT Challenge enabled, the browser appends the string "Authorization:Basic=Uuhjksdkfhk==" to the HTTP GET commands. Since this string contains the PIX authentication credentials and not the IIS authentication credentials, the user is denied access unless the user's AAA username and password match those defined on the Web server.

To get around this issue, the PIX firewall provides a virtual HTTP feature. The Web browser's initial connection is redirected to the virtual HTTP IP address on the PIX firewall. The user is then authenticated, and the browser is redirected to the actual URL that the user requested. Virtual HTTP is transparent to users. To define a virtual HTTP server, use the following command:

```
virtual http <ip_address> [warn]
```

The *ip_address* parameter specifies an unused IP address that is routed to the PIX firewall. The *warn* keyword lets users know that their request was redirected and is only applicable for browsers that cannot redirect automatically.

For example, to enable virtual HTTP using the IP address 10.5.1.15, use the following command:

```
PIX1(config)# virtual http 10.5.1.15
```

Figure 5.34 illustrates the sequence of events that occur when virtual HTTP is enabled.

The steps identified in Figure 5.34 are described here:

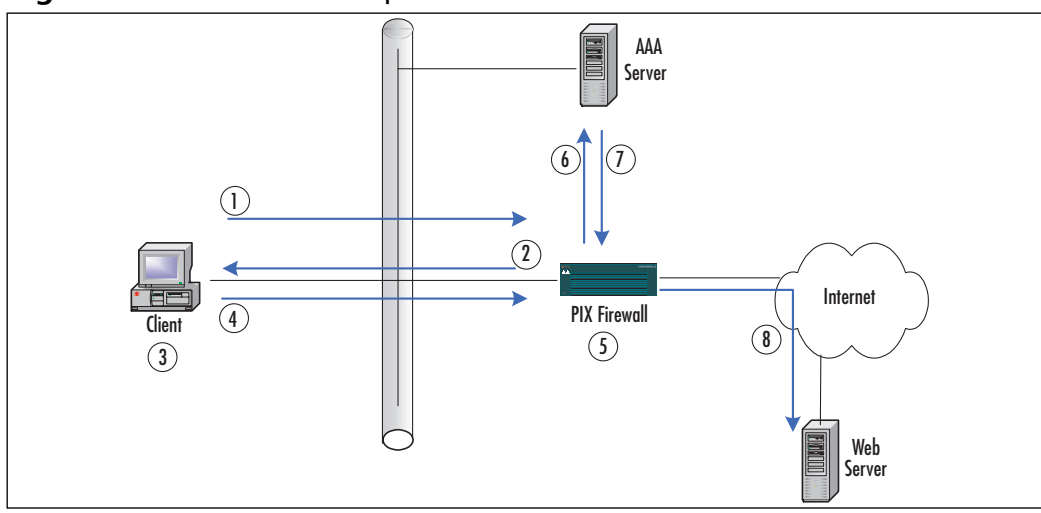
1. The Web browser sends an HTTP request to the Web server.
2. The PIX firewall intercepts the connection attempt and replies with an HTTP 401 Authorization Required response.
3. The Web browser receives the response from the firewall and pops up a dialog box for the user to enter the username and password. The user enters the username and password and presses OK.
4. The Web browser resends the original HTTP request with the username and password embedded as a base64 encoding of "*username*:*password*". The actual field looks similar to the following:

Authorization: Basic ZnJlZDp0aGF0cyBtZQ==

where ZnJlZDp0aGF0cyBtZQ== is the base64 encoded "*username :password*" pair.

5. The PIX firewall receives the HTTP request and splits it into two requests: the AAA authentication request that contains the username and password and the original HTTP request without the username and password.
6. The PIX firewall sends the AAA authentication request to the AAA server.
7. The AAA server attempts to authenticate the user with the provided username and password and sends an ACCEPT or REJECT message.
8. Assuming that the user authenticated successfully, the PIX firewall will then forward the original HTTP request (without the username and password) to the Web server. If the Web server requires its own authentication, it will send its challenge back to the user.

Figure 5.34 Virtual HTTP Operation



With virtual HTTP enabled, once the user has authenticated, he or she will never have to authenticate again as long as there is a Web browser instance active. The *uauth* timer will not expire, because every subsequent Web request will include the encoded and embedded username and password.

WARNING

Do not set the *uauth* timer to 0 if virtual HTTP is enabled, because doing so will prevent connections to the requested (real) Web server.

Use the *show virtual http* command to show the configuration and the *no virtual http* command to disable the use of virtual HTTP.

Virtual Telnet

The *virtual telnet* command has syntax that is similar to the *virtual http* command, but it solves a completely different problem. This feature is useful if you want to preauthenticate users for services that do not support authentication (i.e., services other than HTTP, FTP, or Telnet). Virtual Telnet provides a way for users to authenticate themselves through Telnet before they use those services. For example, let's say that you enabled authentication for all protocols using the *any* keyword as follows:

```
PIX1(config)# aaa authentication include any outbound 0 0 0 0 AuthOut
```

If a user's first outbound connection attempt is anything other than HTTP, FTP, or Telnet, the user will not be able to authenticate and gain access. However, we can configure a virtual Telnet server to preauthenticate the user so they can gain access. This is accomplished using the following command:

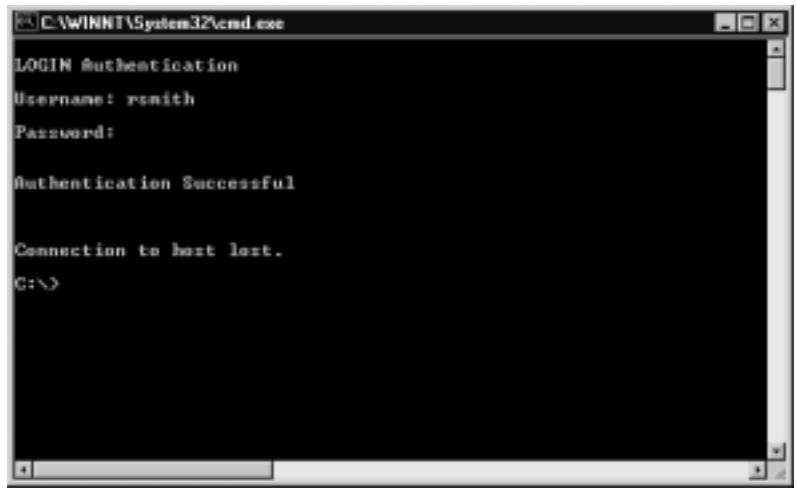
```
virtual telnet <ip_address>
```

The *ip_address* parameter specifies an unused IP address that is routed to the PIX firewall. For example, to enable virtual Telnet on the PIX firewall using IP address 10.5.1.15, use the following command:

```
PIX1(config)# virtual telnet 10.5.1.15
```

The user can now Telnet to the virtual IP address in order to authenticate before using a service that does not support authentication. The user simply Telnets to the IP address of the virtual server and enters his or her AAA username and password. The PIX will then authenticate them, close the Telnet connection, and cache the authentication information for the duration of the *uauth* timer. Figure 5.35 shows an example of authentication using virtual Telnet.

Figure 5.35 A Virtual Telnet Session



You can use virtual Telnet not just for logging in but for logging out as well. After successfully authenticating via virtual Telnet, you will not have to reauthenticate until the *uauth* timer expires. If you are finished with your tasks and want to prevent any further traffic from traversing the firewall using your authentication information, you can Telnet to the virtual IP address again. This effectively ends the session and logs you out.

Use the *show virtual telnet* command to show the configuration and the *no virtual telnet* command to disable the use of virtual Telnet.

Configuring & Implementing…

Changing the Authentication Prompts Used for HTTP, FTP, and Telnet Access

The PIX firewall provides the ability to change the authentication prompts used for HTTP, FTP, and Telnet access. This is accomplished using the following command:

```
auth-prompt [accept | reject | prompt] <string>
```

Set the *accept, reject,* and *prompt* messages appropriately. If you do not specify the *accept, reject,* or *prompt* keywords, the *prompt*

Continued

keyword is assumed. The *string* can be up to 235 alphanumeric characters in length. Spaces and punctuation are allowed, but special characters should not be used. For example:

```
PIX1(config)# auth-prompt prompt Please enter your login credentials
PIX1(config)# auth-prompt accept Authentication Successful
PIX1(config)# auth-prompt reject Authentication Failed
```

To view the authentication prompt configuration, use the *show auth-prompt* command. To remove the configuration, use the *no auth-prompt* command.

Configuring Authorization for Traffic Through the Firewall

Once you have configured authentication for traffic through the firewall using the cut-through proxy, you can also configure *authorization* for traffic through the firewall. Authentication is a requirement for authorization. To implement authorization for traffic through the firewall, you first need to configure the TACACS+ server for authorization.

NOTE

RADIUS and the local database on the PIX firewall are not supported for authorization of traffic through the PIX firewall.

For example, to configure Cisco Secure ACS for authorization of traffic through the PIX firewall, you need to define a shell command authorization set. The section titled "Configuring Cisco Secure ACS to Support TACACS+ Command Authorization" describes how to define a shell command authorization set for authorizing user commands attempted on the firewall itself. The configuration process for services through the firewall is very similar. However, the commands that you enter should be the names of the services that you want to allow (for example, HTTP, Telnet, FTP). If you want to control the destinations that the user can access using the named service, simply enter the desired keyword (*permit* or *deny*) and the IP address in the argument text box. Figure 5.36 provides an example of defining a shell command authorization set for services through the firewall.

Figure 5.36 Defining a Shell Command Authorization Set for Services Through the Firewall

NOTE

Remember that before you can configure a shell command authorization set, you need to configure Cisco Secure ACS to support TACACS+ command authorization.

After configuring the TACACS+ server for authorization, you need to configure AAA authorization on the PIX firewall using the following command:

```
aaa authorization {include | exclude} <author_service> {inbound |
    outbound} [<interface>] <local_ip> <local_mask> <foreign_ip>
        <foreign_mask> <group_tag>
```

The syntax for this command is very similar to that of the *aaa authentication* command. All parameters are the same except for *author_service*. Possible values for the *author_service* parameter are *any, ftp, http, telnet,* or *<protocol/port>*. The possible values for protocol are 6 (TCP), 17 (UDP), 1 (ICMP), and so on. The port value can range from 1 to 65535 and is only valid for the TCP and UDP protocols. Setting the port value to 0 indicates all ports.

For example, the following commands require authorization for all hosts for outbound Telnet, HTTP, and FTP service requests:

```
PIX1(config)# aaa authorization include telnet outbound 0 0 0 0
    AuthOutbound
PIX1(config)# aaa authorization include http outbound 0 0 0 0
    AuthOutbound
PIX1(config)# aaa authorization include ftp outbound 0 0 0 0
    AuthOutbound
```

Configuring Accounting for Traffic Through the Firewall

Accounting can be configured for traffic through the firewall using either RADIUS or TACACS+. It is configured using the following command:

```
aaa accounting {include | exclude} acct_service {inbound | outbound |
    <interface>} <local_ip> <local_mask> <foreign_ip> <foreign_mask>
        <group_tag>
```

NOTE

Accounting can only be configured with RADIUS and TACACS+. There is no such thing as local accounting.

The syntax for this command is very similar to that of the *aaa authentication* command. All parameters are the same except for *acct_service*. Possible values for the *acct_service* parameter are *any, ftp, http, telnet,* or *<protocol/port>*. The possible values for protocol are 6 (TCP) and 17 (UDP), and the port value can range from 1 to 65535. Setting the port value to 0 indicates all ports.

For example, the following command generates accounting data for all hosts that generate any outbound service requests and sends the data to the AAA server in the AuthOutbound group:

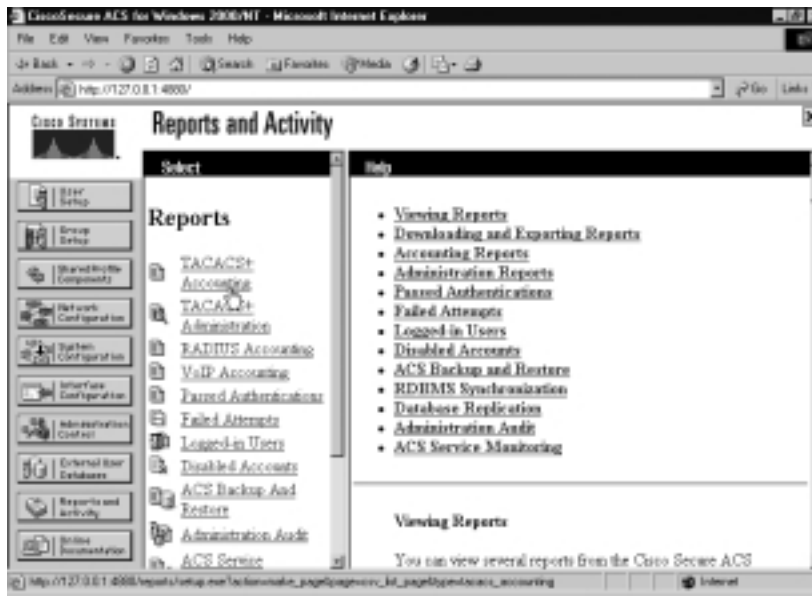
```
PIX1(config)# aaa accounting include any outbound 0 0 0 0 AuthOutbound
```

You do not need to perform any configuration tasks on the Cisco Secure ACS server for it to be able to receive accounting data from a PIX firewall. To view accounting data that is stored on a Cisco Secure ACS server, click the **Reports and Activity** button from the main screen, as shown in Figure 5.37.

Figure 5.37 Cisco Secure ACS: Navigating to Accounting Data

From within the **Reports and Activity** window, click the **TACACS+ Accounting** link, as shown in Figure 5.38.

Figure 5.38 The Cisco Secure ACS Reports and Activity Window



Select the desired TACACS+ accounting file, as shown in Figure 5.39.

Figure 5.39 Cisco Secure ACS: Selecting a TACACS+ Accounting File

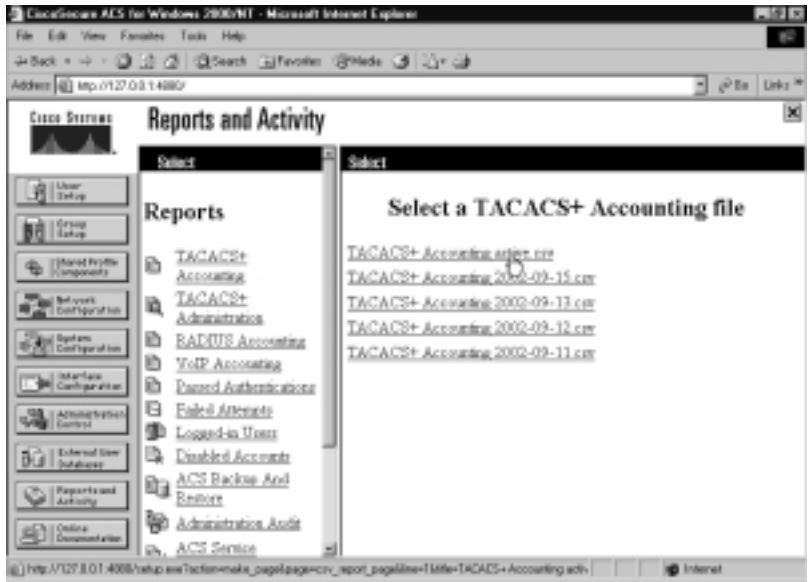
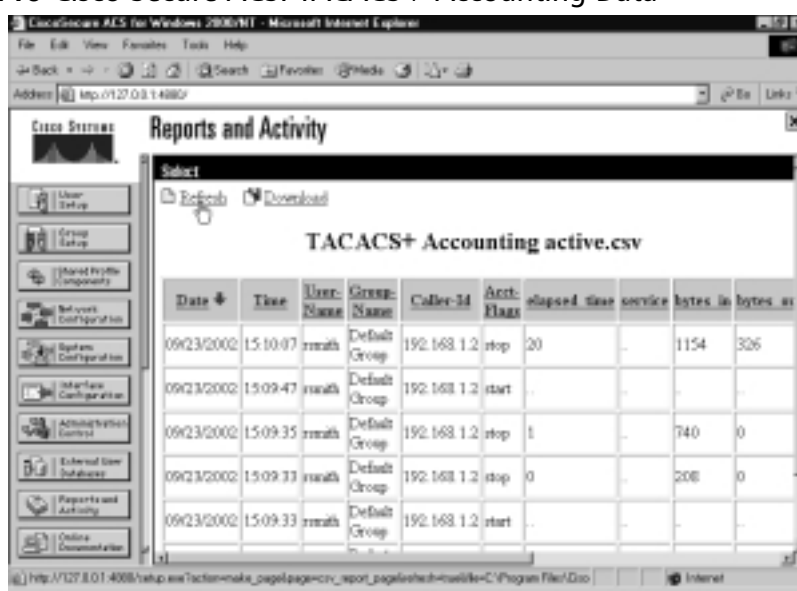


Figure 5.40 shows the type of data that is displayed after you select the desired TACACS+ accounting file.

Figure 5.40 Cisco Secure ACS: TACACS+ Accounting Data



Configuring Downloadable Access Lists

If you need to grant users or groups of users different privileges with respect to services (FTP or HTTP) and hosts that they can access *through* the firewall, the PIX firewall provides the ability to define per-user access lists when used with an AAA server. Unlike earlier versions of the PIX firewall, version 6.2 does not require you to perform any configuration on the PIX firewall itself to implement this capability (assuming that RADIUS authentication and authorization are already configured). You need only to define the desired access-list within the user profile on the Cisco Secure ACS server, and the access list is downloaded to the PIX firewall during user authentication. This simplifies the configuration and improves scalability. There are two options for implementing per-user access lists on the Cisco Secure ACS server:

- **Named downloadable access lists** The PIX firewall downloads a named access list once and can reuse it if you have assigned it to other users. You should use named access lists if you have multiple users that share an access list or if you have a large access list that is assigned to more than one user.
- **Unnamed downloadable access lists** The PIX firewall downloads an unnamed access list for each user to which you assigned one. These access lists are not shared and are downloaded each time a user is authenticated. You should use an unnamed access list if a different access list is defined for every user.

NOTE

Downloadable ACLs are supported only with RADIUS, not TACACS+.

Configuring Named Downloadable Access Lists

Named downloadable access lists are shared profile components within Cisco Secure ACS. Shared profile components are reusable authorization definitions that need to be created only once and can be shared among users and groups. In other words, instead of having to recreate a given access list every time you add a new user, you can created the access list once and then apply it to users as they are added. This eases the administrative burden significantly and increases the scalability of the authorization controls.

You need to complete two main tasks to configure named downloadable access lists within Cisco Secure ACS:

1. Define the named downloadable access list within the **Shared Profile Components** section of Cisco Secure ACS.
2. Apply the named downloadable access list to the appropriate users within the **User Setup** section of Cisco Secure ACS.

To define a named downloadable access list, click the **Shared Profile Components** button on the left side of the Cisco Secure ACS HTML interface, as shown in Figure 5.41.

Figure 5.41 Cisco Secure ACS Main Interface User Setup

Click **Downloadable PIX ACLs** in the Shared Profile Components window, as shown in Figure 5.42.

Within the Downloadable PIX ACLs window, click the **Add** button to define a new downloadable access-list, as shown in Figure 5.43.

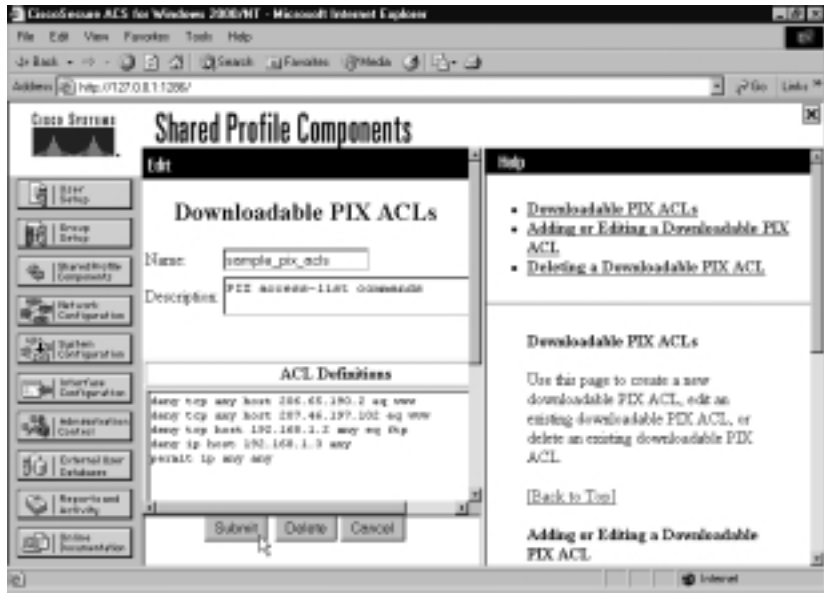
The Downloadable PIX ACLs Edit window, shown in Figure 5.44, allows you to define a new downloadable access list. Enter a name for the access list in the Name text box and an optional description in the Description text box. In the ACL Definitions text box, enter the entries for the access list. Create an entry using the syntax of the *access-list* command, omitting both the *access-list* keyword and the name of the access list.

Figure 5.42 The Cisco Secure ACS Shared Profile Components Window

Figure 5.43 The Cisco Secure ACS Downloadable PIX ACLs Window

Figure 5.44 shows an example. When you finish creating the access list entries, click the **Submit** button.

Figure 5.44 The Cisco Secure ACS Edit Downloadable ACLs Window



Now that you have defined the named access list, it is available for you to assign to users. Click the **User Setup** button on the left side of the Cisco Secure ACS HTML interface and select a user that you want to edit (or add a new user as shown previously in the section "Adding a New User to Cisco Secure ACS"). Scroll down within the User Setup window until you see the Downloadable ACLs section, as shown in Figure 5.45. Select the **Assign PIX ACL** check box, and select the appropriate named access list from the corresponding drop-down list. Click the **Submit** button to assign the access list to the user.

NOTE

If you do not see the Downloadable ACLs section, you need to enable this option by clicking the **Interface Configuration** button from the Cisco Secure ACS main screen, clicking **Advanced Options**, then selecting the **User-Level Downloadable ACLS** and **Group-Level Downloadable ACLS** check boxes.

Figure 5.45 Cisco Secure ACS: Assigning Downloadable ACL

You do not have to configure anything on the PIX firewall to complete the downloadable access list configuration. When a user authenticates to the PIX firewall, the access list will be downloaded with a name that has the following format:

```
#ACSACL#-<acl_name>-<version_id>
```

In this syntax, *acl_name* is the name that you gave the access list within Cisco Secure ACS, and *version_id* is a unique ID assigned to the access list. Figure 5.46 provides an example of what the downloadable access list looks like on the PIX firewall. Within the figure, the first *show access-list* command was issued *before* user authentication, and the second *show access-list* command was issued *after* user authentication. As you can see, no access lists were defined before user authentication, but there is a downloadable access list defined after authentication, and it has a name that complies with the format identified previously.

Figure 5.46 Named Downloadable Access List: PIX Firewall View

```
PIX1(config)# show access-list
PIX1(config)# show access-list
access-list #ACSACL#-PIX-sample-pix_acls-3d7fe64e; 5 elements
access-list #ACSACL#-PIX-sample-pix_acls-3d7fe64e deny tcp any host
    206.65.190.2 eq www (hitcnt=0)
```

Continued

Figure 5.46 Continued

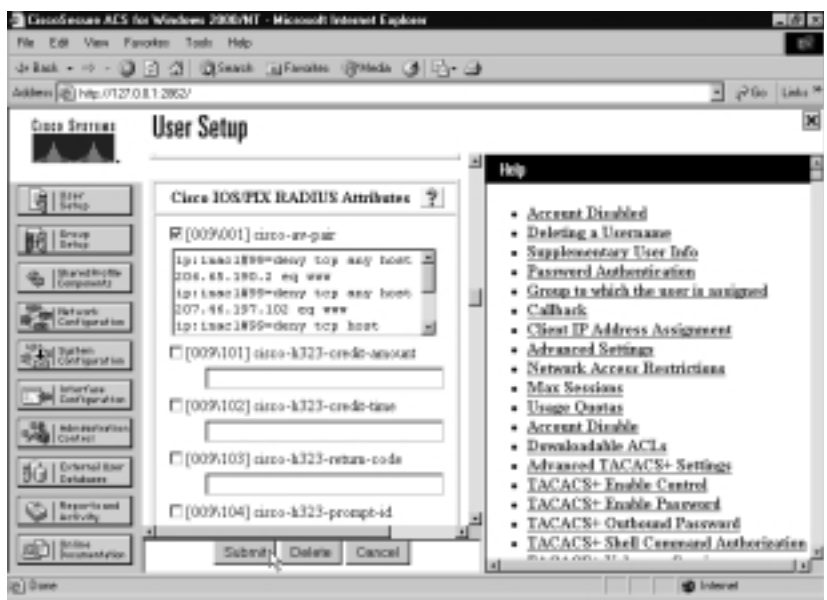
```
access-list #ACSACL#-PIX-sample-pix_acls-3d7fe64e deny tcp any host
     207.46.197.102 eq www (hitcnt=0)
access-list #ACSACL#-PIX-sample-pix_acls-3d7fe64e deny tcp any host
     192.168.1.2 any eq ftp (hitcnt=0)
access-list #ACSACL#-PIX-sample-pix_acls-3d7fe64e deny ip host
     192.168.1.3 any (hitcnt=0)
access-list #ACSACL#-PIX-sample-pix_acls-3d7fe64e permit ip any any
     (hitcnt=2)
```

Configuring Downloadable Access Lists Without Names

To configure downloadable access lists without names, navigate to the selected user within the User Setup window, and scroll down to the Cisco IOS/PIX RADIUS Attributes section of the window. As shown in Figure 5.47, select the **[009\001] cisco-av-pair** check box and make the desired access list entries in the text box. The entries should have the following format:

```
ip:inacl#<n>=<acl_entry>
```

Figure 5.47 Cisco Secure ACS: User Setup—Cisco IOS/PIX RADIUS Attributes



The *ip:inacl#* keyword specifies a number (*n*) between 0 and 999999999 that identifies the order of the *access-list* entry. The *acl_command* parameter is an access list entry statement without the *access-list* command or the name of the access list.

NOTE

If you do not see the Cisco IOS/PIX RADIUS attributes displayed within the user setup, you need to enable them via the Interface Configuration window.

Figure 5.48 provides an example of what the unnamed downloadable access list looks like on the PIX firewall. Within the figure, the first *show access-list* command was issued *before* user authentication, and the second *show access-list* command was issued *after* user authentication. As you can see, no access lists were defined before user authentication, but there is a downloadable access list defined after authentication.

Figure 5.48 Unnamed Downloadable Access List: PIX Firewall View

```
PIX1(config)# show access-list
PIX1(config)# show access-list
access-list AAA-user-rsmith; 5 elements
access-list AAA-user-rsmith deny tcp any host 206.65.190.2 eq www
     (hitcnt=0)
access-list AAA-user-rsmith deny tcp any host 207.46.197.102 eq www
     (hitcnt=0)
access-list AAA-user-rsmith deny tcp any host 192.168.1.2 any eq ftp
     (hitcnt=0)
access-list AAA-user-rsmith deny ip host 192.168.1.3 any (hitcnt=0)
access-list AAA-user-rsmith permit ip any any (hitcnt=4)
```

Summary

This chapter provided an overview of AAA and its benefits and described the RADIUS and TACACS+ security protocols. AAA comprises the three independent but related functions of authentication, authorization, and accounting, which are defined as follows:

- *Authentication* is the process of identifying and authenticating a user before allowing access to network devices and services. User identification and authentication are critical for the accuracy of the authorization and accounting functions.
- *Authorization* is the process of determining user privileges and access rights after users have been authenticated.
- *Accounting* is the process of recording user activities for accountability, billing, auditing, or reporting purposes.

The benefits of implementing AAA include scalability, increased flexibility and control, standardized protocols and methods, and redundancy. Cisco PIX firewalls support the RADIUS and TACACS+ security protocols for use within an AAA mechanism. Each protocol has its advantages and disadvantages; the protocol that is right for you will depend on your situation and requirements.

To take advantage of AAA, you must implement and configure an AAA server. Cisco Secure Access Control Server (ACS) is AAA server software that simultaneously supports both the TACACS+ and RADIUS protocols. After installing the software, you can perform basic tasks such as adding users AAA clients. In addition, you can perform advanced tasks such as defining downloadable access lists and command authorization sets.

On the PIX firewall, you can configure authentication and authorization to control both user actions *on* the firewall and user actions *through* the firewall. Authentication of users attempting to access the PIX firewall itself is called *console authentication*. Authorization of user actions *on* the PIX firewall is called *command authorization*. For both console authentication and command authorization, you can use the local database, RADIUS, or TACACS+.

For user actions *through* the PIX firewall, Cisco provides a feature called *cut-through proxy* to support user authentication and authorization. Cut-through proxy allows you to implement authentication and authorization for inbound or outbound HTTP, FTP, and Telnet connections. This functionality allows you to control services available through the firewall by user identity rather than IP address, giving you a finer granularity of control. Because cut-through proxy only

authenticates and authorizes the initial connection attempt, it provides performance advantages over traditional proxy firewalls because subsequent communication occurs directly between the two endpoints while being inspected by the firewall.

Virtual HTTP and virtual Telnet are features related to cut-through proxy. Virtual HTTP solves an authentication issue that exists for some Microsoft IIS servers that have Basic Authentication or NT Challenge enabled. Virtual Telnet provides a mechanism for users to preauthenticate to the PIX firewall before using services that do not support authentication.

Downloadable ACLs allow you to configure per-user or per-group access lists centrally on the AAA server, thereby decreasing administrative overhead and increasing scalability.

Solutions Fast Track

AAA Concepts

- ☑ AAA is an architectural framework composed of the three independent but related functions of authentication, authorization, and accounting. The benefits of implementing AAA include scalability, increased flexibility and control, standardized protocols and methods, and redundancy.
- ☑ Authentication is the process of identifying and authenticating a user before allowing access to network devices and services.
- ☑ Authorization is the process of determining a user's privileges and access rights after they have been authenticated.
- ☑ Accounting is the process of recording user activities for accountability, billing, auditing, or reporting purposes.

Cisco Secure ACS for Windows

- ☑ To take advantage of AAA, you must implement and configure an AAA server. Cisco Secure Access Control Server (ACS) is AAA server software that supports both the TACACS+ and RADIUS protocols.
- ☑ Cisco Secure ACS includes its own internal database, but it also supports authentication against the following external user databases: Windows NT/2000, Generic LDAP, Novell NetWare Directory Services (NDS),

Open Database Connectivity (ODBC)-compliant relational databases, CRYPTOCard token server, SafeWord token server, AXENT token server, RSA SecureID token server, ActivCard token server, and Vasco token server.

Configuring Console Authentication

- ☑ Console authentication is used to authenticate users attempting to access the PIX firewall itself. It can be configured to use the LOCAL, TACACS+, or RADIUS databases.
- ☑ To use local console authentication, you need to define users on the PIX firewall using the *username* command.
- ☑ To use TACACS+/RADIUS console authentication, you need to perform configuration tasks on the TACACS+/RADIUS server. You need to define the PIX firewall as an AAA client to the server and create user accounts on the server.

Configuring Command Authorization

- ☑ Command authorization controls user actions on the PIX firewall. It can use the LOCAL or TACACS+ databases.
- ☑ To use local command authorization, you need to define users on the PIX firewall using the *username* command and assign commands to selected privilege levels using the *privilege* command.
- ☑ To use TACACS+ command authorization, you need to define command authorization sets on the TACACS+ server and assign these command authorization sets to users.

Configuring Authentication for Traffic Through the Firewall

- ☑ Cut-through proxy allows you to perform user authentication and authorization of user actions *through* the PIX firewall. Specifically, it allows you to implement authentication and authorization for inbound or outbound HTTP, FTP, and Telnet connections and allows you to

control services available through the firewall by user identity rather than IP address, which gives you a finer granularity of control.

- ☑ Because the cut-through proxy only authenticates and authorizes the initial connection attempt, it provides performance advantages over traditional proxy firewalls because subsequent communication occurs directly between the two endpoints while being inspected by the firewall.
- ☑ You can control how frequently cut-through proxy users need to reauthenticate by setting inactivity and absolute *uauth* timers.
- ☑ With cut-through proxy authentication enabled for Web traffic (i.e., HTTP), your users could experience some problems when connecting to Web sites that run Microsoft IIS with Basic Authentication or NT Challenge enabled. The PIX firewall gets around this issue by providing a virtual HTTP feature. Once enabled, the PIX firewall will redirect incoming HTTP requests that require authentication to the virtual server IP address, authenticate the user, then redirect the browser back to its original requested destination.
- ☑ If you enabled AAA authentication for services that do not support authentication (i.e., services other than HTTP, FTP, or Telnet), virtual Telnet provides a way for users to preauthenticate themselves before they use those services.

Configuring Authorization for Traffic Through the Firewall

- ☑ Once you have configured authentication for traffic through the firewall using the cut-through proxy, you can also configure authorization for traffic *through* the firewall.
- ☑ To configure Cisco Secure ACS for authorization for traffic through the PIX firewall, you need to define a shell command authorization set. You define a shell command authorization set for authorizing traffic through the firewall in the same manner that you do for command authorization; however, the commands that you enter should be the name of the service that you want to allow (e.g., HTTP, Telnet, FTP).

Configuring Accounting for Traffic Through the Firewall

- ☑ You do not need to perform any configuration tasks on the Cisco Secure ACS server for it to be able to receive accounting data from a PIX firewall.
- ☑ To view accounting data that is stored on a Cisco Secure ACS server, click the **Reports and Activity** button from the main screen, click the **TACACS+ Accounting** link, and select the desired TACACS+ accounting file.

Configuring Downloadable Access Lists

- ☑ If you need to grant users or groups of users different privileges with respect to the services and hosts that they can access through the firewall, the PIX firewall provides the capability to define per-user access lists when used with an AAA server.
- ☑ Named downloadable access lists can be defined on Cisco Secure ACS and shared among users and groups. Instead of having to recreate a given access list every time you add a new user, you can created the access list once and then apply it to users as they are added.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: Are there AAA protocols other than RADIUS and TACACS+?

A: Yes. We identified and briefly discussed TACACS and XTACACS, which are no longer supported by Cisco and are not used much anymore. In addition, DIAMETER is an AAA protocol that is designed to coexist with RADIUS. It is still under development by the IETF's AAA Working Group. You can find more information at www.diameter.org.

Q: I am interested in implementing a RADIUS server. Where can I find information on RADIUS products?

A: While you can certainly perform a Web search to identify RADIUS products, a good listing can be found at http://ing.ctit.utwente.nl/WU5/D5.1/Technology/radius/index.html#products. In addition, a listing of TACACS+ products can be found at http://ing.ctit.utwente.nl/WU5/D5.1/Technology/tacacs+/index.html#products.

Q: I am new to configuring the PIX firewall and am unsure if I have configured AAA correctly. Is there a way that I can check my configuration?

A: Output Interpreter, a tool on the Cisco support Web site, can analyze your PIX configuration and will report errors, potential problems, and suggested fixes. You simply select PIX from the drop-down list and highlight the **show terminal** selection. Enter the *show terminal* command on your PIX, paste the output into the text box, and click the **Submit** button. Output Interpreter will analyze the configuration and provide you feedback. The tool is located at www.cisco.com/cgi-bin/Support/OutputInterpreter/home.pl.

Q: Does the PIX firewall support AAA for authenticating Cisco software VPN clients?

A: Yes. The PIX provides support for AAA authentication with Cisco VPN clients using *xauth*. You can find more information on this topic in Chapter 7.

Chapter 6

Configuring System Management

Solutions in this chapter:

- **Configuring Logging**
- **Configuring Remote Access**
- **Configuring Simple Network Management Protocol**
- **Configuring System Date and Time**

☑ **Summary**

☑ **Solutions Fast Track**

☑ **Frequently Asked Questions**

Introduction

System management is an important part of configuring and maintaining your firewall. Without proper management, security policies cannot be enforced or monitored and a device might be compromised. In this chapter, we focus on managing individual PIX firewalls.

Logging is important, but not just for monitoring or troubleshooting; it is invaluable for measuring system performance, identifying potential network bottlenecks, and in today's brave new security-conscious world, detecting potential security violations. In this chapter, you will learn how to enable and customize local and remote logging. Remote administration is another important component of system management. You will learn how to configure a variety of in-band management protocols, such as SSH, Telnet, and HTTP, to remotely configure and monitor the PIX firewall. We will discuss the security implications of each protocol and situations in which one protocol might be more appropriate than another. You will also learn how to use out-of-band management using SNMP. We will discuss configuring the system date and time and why it plays a vital role in system management. Along with system date and time, you will learn how to use NTP to make easier the job of managing accurate and consistent time and date across multiple devices.

Configuring Logging

Logging is one of the most important yet least understood methods of managing the Cisco PIX firewall. Logging offers a wealth of information about what is happening on the PIX, who is doing what, who is going where, and possible attacks or probes. Rumor has it that logging is very complicated and cumbersome to do, but in reality, it is not that hard.

The Cisco PIX firewall provides a significant amount of logging functionality. However, all logging is disabled by default. It is up to you to decide how much or how little logging to enable, configure, and use. On the Cisco PIX, there are two ways to log information: local and remote. Local logging is of limited archival value, so it is highly recommended that remote logging be used to gather information. Remote logging of messages allows you to store the messages and use scripts to examine the messages in detail, manipulate the data, and generate detailed reports. Remote message logging also lets you archive events and keep a historical record. For remote logging, the PIX firewall uses syslog, which is a traditional UNIX method of logging and is described in RFC 3164. The remote

logging server (known as the *syslog server*) can be based on the Windows, Linux/UNIX, or Macintosh platform. In this chapter, we focus on Windows and Linux/UNIX syslog servers.

Logging on the PIX firewall can be performed at one of several levels of detail. Level 3 (error) is the default for the PIX. Level 7 (debug) is the most verbose and is recommended only when you are troubleshooting the PIX. In normal network operations, Cisco recommends using Level 4 (warning) or Level 3 (error).

In the course of normal logging (Level 3), the PIX firewall logs alerts (such as a failover link going down), error conditions (such as ICMP being blocked), and informational messages (such as a memory allocation error). If configured for a higher logging level, the PIX firewall logs connection setup and teardown, as well as the amount of traffic transferred in each session. This functionality can be useful if you are trying to gather statistics on how much traffic is being exchanged per protocol or per session.

It is possible to view logging messages in real time, either through a Telnet or SSH session or on the console port. Both methods carry a risk of being overwhelmed by messages, depending on the logging level. A Telnet or SSH session can time out and drop the session, and the console port can lock up to the point where you cannot type in the command to turn logging off. You must use caution when viewing log messages using these methods.

Local Logging

The PIX offers both local and remote message logging. Normally, remote logging is preferred over local logging, but when you're troubleshooting or configuring the Cisco PIX firewall, it can be useful to have local logging enabled. Three types of logging are available locally: buffered logging, console logging, and terminal logging. Since logging on the PIX is disabled by default, you need to enable it using the following command:

```
PIX1(config)# logging on
```

This command is required to start logging to all output locations such as the buffer, console, terminal, or syslog server. However, after entering this command, you still must specify the individual logging methods. To disable logging, use the *no* form of the command:

```
PIX1(config)# no logging on
```

Buffered Logging

The first method of local logging we discuss is known as *buffered logging*. When you use this method, all log messages are sent to an internal buffer on the PIX firewall. To enable buffered logging, use the following command:

```
PIX1(config)# logging buffered <level>
```

The *level* parameter specifies the level of detail you want to see in your logs. (Logging levels are discussed later in this chapter.) To view the messages held in the buffer, use the following command:

```
show logging
```

This command shows the logging configuration as well as the messages that are held in the log buffer. The PIX firewall can only log up to 100 messages to the log buffer, so it usually is not necessary to clear this buffer. However, if you choose, you can use the *clear logging* command in Enable mode to clear out the buffer and start fresh. In order to disable buffered logging, use the *no logging buffered* command in configuration mode. Here is an example of the *show logging* command:

```
PIX1# show logging
Syslog logging: enabled
    Facility: 20
    Timestamp logging: disabled
    Standby logging: disabled
    Console logging: level debugging, 37 messages logged
    Monitor logging: disabled
    Buffer logging: level debugging, 9 messages logged
    Trap logging: disabled
    History logging: disabled
111008: User 'enable_15' executed the 'logging buffered 7' command.
111009: User 'enable_15' executed cmd: show logging
```

This command clearly shows the logging configuration in detail as well as the contents of the log buffer. We can see the types of logging that are enabled and the number of messages logged. In this example, console and buffered logging are enabled (both at level debugging). There are also two messages in the logging buffer in this example. We discuss console logging next.

Console Logging

When enabled, *console logging* sends log messages to the console (serial port) of the PIX firewall. To enable console logging, use the following command:

```
PIX1(config)# logging console <level>
```

The *level* parameter has the same meaning as discussed previously. Once entered, logging messages are printed to the console. If there are too many messages, it can be very distracting to try to type in commands while messages are being printed to the screen. Furthermore, logging more than you need can degrade the performance of the PIX firewall. To stop the printing of messages to the console, use the following command:

```
PIX1(config)# no logging console
```

Terminal Logging

Terminal logging sends log messages to a Telnet or SSH session. To enable terminal logging, use the following command:

```
PIX1(config)# logging monitor <level>
```

In addition to enabling this function at a global level, logging output must be enabled on a per-session basis. To enable the display of syslog messages in the current Telnet or SSH session, use the following command:

```
PIX1# terminal monitor
```

When you no longer want to view log messages in your Telnet or SSH session, you can disable monitoring using the *terminal no monitor* command in Enable mode. It is possible to lose control of your Telnet or SSH session when too much data is being printed to the screen. You can recover by restarting your Telnet or SSH session.

Syslog

Syslog is one of the most common methods for capturing and saving log messages. In order for syslog to work, you need to configure the host that will send the syslog messages as well as the syslog server, which will receive the syslog messages. In our case, the PIX firewall will be the host sending the log messages to a syslog server, which can be Linux/UNIX, Windows, or even Macintosh based.

The syslog server determines where to place the log messages. Depending on which syslog server software is being used and how it is configured, the syslog

server may write the messages to a file or send an alert to an engineer by e-mail or pager.

On a typical enterprise network, depending on the configured logging level, a busy PIX firewall can log messages to use up several gigabytes of space a day on the syslog server. A prudent engineer will set storage limits on his syslog server (usually in megabytes) and configure it to overwrite older messages as needed, thus ensuring that available storage space is not overrun.

As described previously, since logging on the PIX is disabled by default, you need to enable it:

```
PIX1(config)# logging on
```

To configure syslog on the PIX, you first need to tell the firewall which host to send the syslog messages to. To do this, use the following command:

```
logging host [<interface>] <ip_address>
```

The *interface* parameter specifies the interface you want to send the messages out on, and the *ip_address* parameter specifies the IP address of the syslog server on that interface. If not specified, the interface is assumed to be the inside interface. No log messages will be sent to syslog until you configure the logging level using the following command:

```
logging trap <level>
```

The *level* parameter specifies the severity level, as discussed later in this chapter.

Here is an example of configuring syslog on the PIX firewall:

```
PIX1(config)# logging host inside 192.168.50.8
PIX1(config)# logging trap debugging
PIX1(config)# logging on
PIX1(config)#
PIX1# show logging
Syslog logging: enabled
    Facility: 20
    Timestamp logging: disabled
    Standby logging: disabled
    Console logging: disabled
    Monitor logging: disabled
    Buffer logging: disabled
    Trap logging: level debugging, 38 messages logged
```

```
        Logging to inside 192.168.50.8
    History logging: disabled
```

In this example, logging is configured to send messages to the syslog server 192.168.50.8 on the inside interface with a severity level of debugging.

When configured to use syslog, the PIX firewall will send the log messages to the syslog server using UDP port 514 by default. You can change this default behavior by entering the longer form of the *logging host* command:

```
logging host [<interface>] <ip address> [tcp|udp/<port_number>]
```

You can configure either UDP or TCP for syslog, and the *port_number* parameter can be any value from 1025 to 65535. TCP is not a standard method for handling syslog, and most servers do not support it, but it can provide reliable logging. If you will be using a TCP connection to the syslog server, there is an important warning to remember: If the syslog server goes down when you're using TCP, the default behavior for the PIX firewall is that all network traffic through the PIX will be *blocked*. Also important to remember when configuring TCP syslog is that the syslog connection will be slower than UDP since TCP relies on the three-way handshake to start a connection and each packet must be acknowledged. This will add to the overhead of the connection and slow the sending of syslog messages to the server.

In the following example, we configure syslog using TCP. The *port_number* parameter has been set to 1468, which is the default TCP port used by syslog servers that accept TCP syslog from PIX firewalls. Do not forget to configure the syslog server to listen on TCP port 1468 for syslog messages.

```
PIX1(config)# logging host inside 192.168.50.9 tcp/1468
PIX1(config)# logging trap debugging
PIX1(config)# logging on
PIX1(config)#
PIX1# show logging
Syslog logging: enabled
    Facility: 20
    Timestamp logging: disabled
    Standby logging: disabled
    Console logging: disabled
    Monitor logging: disabled
    Buffer logging: disabled
    Trap logging: level debugging, 31 messages logged
```

```
        Logging to inside 192.168.50.9 tcp/1468
    History logging: disabled
```

Although the PIX firewall can have multiple logging hosts configured, it can only use a single protocol with each logging host. In the event that your syslog server is offline, the PIX will start to queue the syslog messages in memory and then start to overwrite the held messages, starting with the oldest first. The following command is used to configure the size of the syslog message queue in memory:

```
logging queue <msg_count>
```

The default is 512 messages. The *msg_count* parameter specifies the size of the syslog message queue. If *msg_count* is set to 0, the queue size is unlimited and based on the available block memory.

To see the queue statistics and any discarded message statistics, use the following command:

```
PIX1# show logging queue

        Logging Queue length limit : 512 msg(s)
        Current 3 msg on queue, 5 msgs most on queue
```

One of the Cisco PIX firewall's features is the ability to have a failover PIX. One logging command allows the failover PIX to send syslog messages in order for the log files to be synchronized in the case of stateful failover taking place. This command, *logging standby*, is disabled by default since it will double the amount of syslog traffic on the network. Once it is enabled, you can turn off this command using the *no logging standby* command.

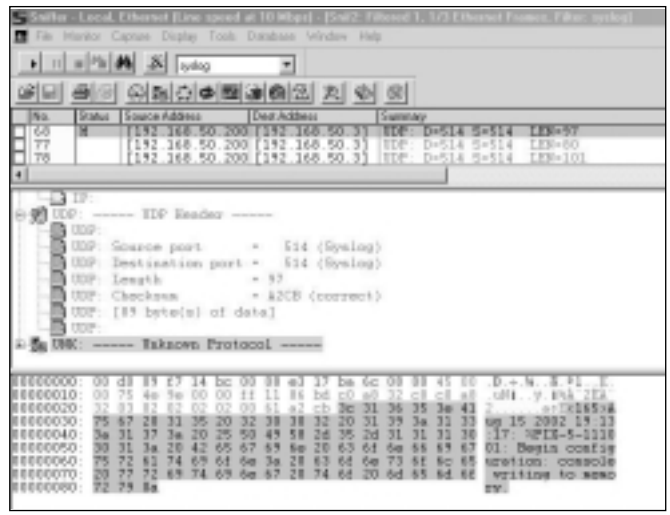
To ensure that the syslog messages are sent to the syslog server with a timestamp, configure the *logging timestamp* command in configuration mode. This command requires that the *clock* command be set on the PIX. To turn off timestamps, use the *no logging timestamp* command in Configuration mode.

NOTE

Sometimes it is forgotten that the syslog messages are sent in plain text and should not be considered secure. In Figure 6.1, we can see a Sniffer capture, which shows that the log message is in clear text. If you are sending log files across WAN links or the Internet or have a need for high

security, you should consider using an encrypted link to transmit syslog messages.

Figure 6.1 Sniffer Syslog Trace



Many syslog server applications are available for both Microsoft Windows and Linux/UNIX. Virtually all Linux/UNIX systems have syslog enabled for logging local messages, and it only takes some minor adjustments to enable remote logging. Microsoft Windows requires that a syslog server be installed since syslog services are not a part of the operating system. One popular choice for a Microsoft Windows syslog server is the Kiwi Syslog Daemon, available at www.kiwisyslog.com, which runs on all versions of Windows, from Windows 98 to Windows XP. With Windows NT and Windows 2000, the Kiwi Syslog Daemon can be installed as a service. The Kiwi Syslog Server can be configured to use either UDP or TCP to accept syslog messages from the Cisco PIX firewall. Figure 6.2 shows the Kiwi Syslog Daemon default log screen.

Another choice is the free PIX Firewall Syslog Server (PFSS) available from Cisco. If you have access to Cisco CCO, you can download PFSS at www.cisco.com/cgi-bin/tablebuild.pl/pix. PFSS is a very basic syslog server for Windows that can use either UDP or TCP syslog with the Cisco PIX firewall. PFSS runs under Windows NT as a service and does not support Windows 95, Windows 98, or Windows ME. Unlike Kiwi Syslog Server, the PFSS message log file is very basic in presentation, as shown in Figure 6.3.

Figure 6.2 The Kiwi Syslog Server Default Log Screen

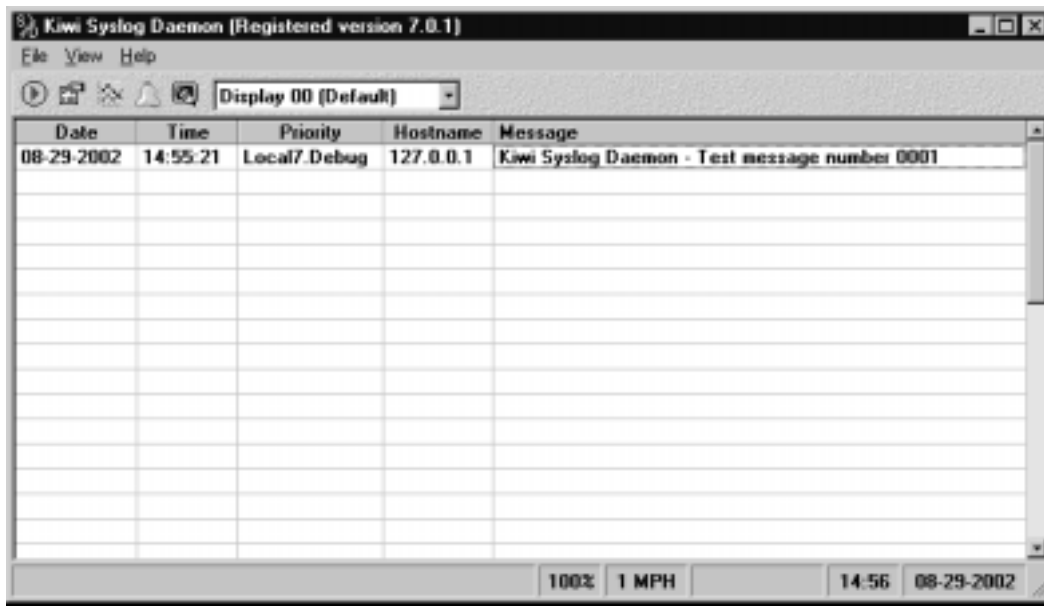
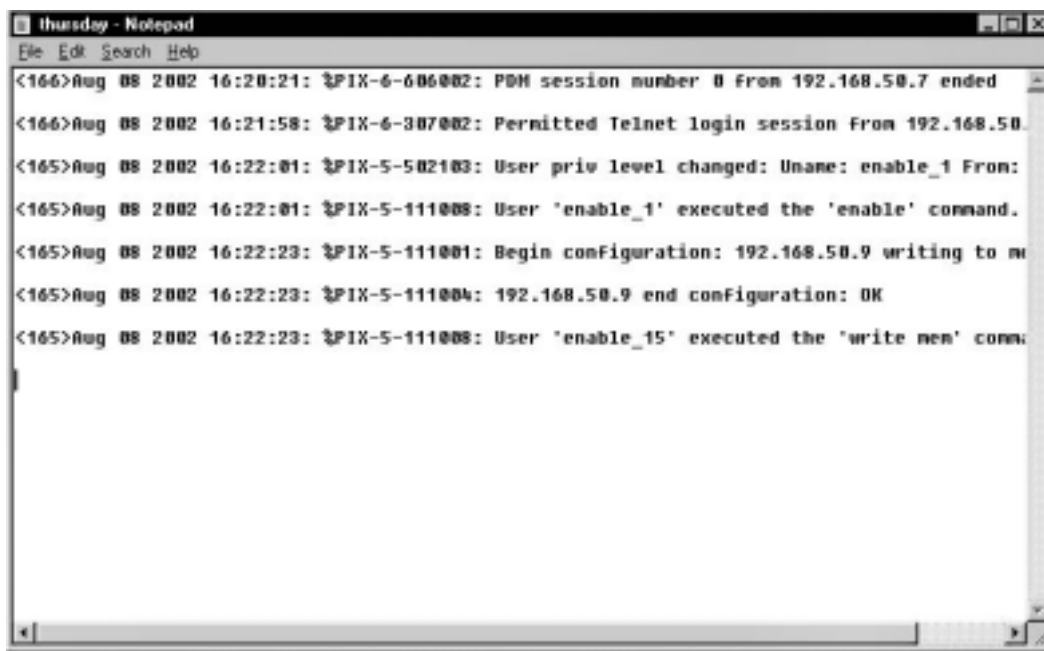


Figure 6.3 The Cisco PFSS Log File



In the world of Linux/UNIX, syslog is normally a service or daemon that has been installed by default to provide local message logging. Some minor configuration changes might need to be made to enable remote syslog functions. The daemon that controls syslog on Linux/UNIX is called *syslogd*. This daemon is part of the normal startup of a Linux box. In the figures that follow, we use RedHat 7.1 as the Linux server.

The first requirement is to reconfigure syslogd to accept remote syslog messages. Log into the Linux machine with proper permissions and then use the *ps* command to verify that syslogd is running:

```
linux1# ps -ef | grep syslogd
root       2000      1   0 22:03 ?          00:00:00 syslogd -m 0
```

As you can see from the output of this command, on this particular machine the syslog daemon is running and has a process ID of 2000. In order for the Linux syslog daemon to accept messages from remote hosts, the syslog configuration needs to be changed by adding -r to the startup configuration. This is accomplished by editing the /etc/sysconfig/syslog file and adding -r to the SYSLOGD_OPTIONS so that it looks like this:

```
SYSLOGD_OPTIONS="-m 0 -r"
```

We will now restart the syslog daemon by using the following command:

```
linux1# /etc/rc.d/init.d/syslog restart
```

When syslogd has restarted, you should verify that it is running by issuing the *ps* command again:

```
linux1# ps -ef | grep syslogd
root       2160      1   0 22:05 ?          00:00:00 syslogd -m 0 -r
```

The system should now be ready to accept syslog messages from remote hosts.

Logging Levels

Although the *logging* command has eight different severity levels that are used on the PIX (Levels 0 through 7), logging Level 0 (emergency) is not used. It is only represented for compatibility with UNIX syslogging. When you configure logging, you must specify a severity level by a number or keyword. When you specify a level, the PIX firewall logs all events equal to the specified level as well as the levels below it. For example, the default severity level for the PIX is 3 (error), which also logs Level 2 (critical), Level 1 (alert), and Level 0 (emergency) events. A complete list of the keywords and equivalent levels is shown in Table 6.1.

Table 6.1 Logging Levels and Messages

Keyword	Level	Message
emergency	0	System unusable
alert	1	Immediate action needed
critical	2	Critical condition
error	3	Error condition
warning	4	Warning condition
notification	5	Normal but significant condition
informational	6	Informational message only
debugging	7	Only used during debugging

A system log message that the syslog server will receive is structured like this:

```
%PIX-Level-message_number: Message_text
```

The syslog messages will be prefaced with a time and date stamp and the source IP address. This will be followed by the *Level*, which represents the logging level of the message. For example, the message snippet *%PIX-2-106016:* shows us that the logging level for this message is 2 (critical). The *message_number* is a numeric code that is unique for the type of message. This example of *106016* is for the message "Deny IP spoof from (*IP_addr*) to *IP_addr* on interface *int_name.*" When you configure the PIX to disable certain messages, you will use the numeric code to identify which message to disable.

Here are some sample messages at the various logging levels:

- **Level 1**

```
%PIX-1-101002: (Primary) Bad fail over cable.
%PIX-1-101003: (Primary) Fail over cable not connected (this unit)
```

- **Level 2**

```
%PIX-2-106016: Deny IP spoof from (IP_addr) to IP_addr on interface
    int_name.
%PIX-2-106017: Deny IP due to Land Attack from IP_addr to IP_addr.
```

- **Level 3**

```
%PIX-3-201005: FTP data connection failed for IP_addr
%PIX-3-201008: The PIX is disallowing new connections.
```

- **Level 4**

```
%PIX-4-403110: PPP virtual interface int_name, user: user missing
    MPPE key from aaa server.
%PIX-4-404101: ISAKMP: Failed to allocate address for client from
    pool pool_id
```

- **Level 5**

```
%PIX-5-500001: ActiveX content modified src IP_addr dest IP_addr on
    interface int_name.
%PIX-5-500002: Java content modified src IP_addr dest IP_addr on
    interface int_name.
```

- **Level 6**

```
%PIX-6-109005: Authentication succeeded for user 'user' from
    laddr/lport to faddr/fport on interface int_name.
%PIX-6-109006: Authentication failed for user 'user' from
    laddr/lport to faddr/fport on interface int_name.
```

- **Level 7**

```
%PIX-7-702301: lifetime expiring
%PIX-7-702303: sa_request
```

The Cisco PIX firewall has the ability to log URL and FTP requests. URL logging catches the URL's IP address and the names of any accessed files. FTP logging shows the IP address that is being accessed, the actions performed (file retrieved or stored), and the names of the files that were transferred. To enable URL logging, enable fixup for HTTP, set the logging level to 5 (notification), and look for the message type 304001. For example:

```
%PIX-5-304001: 192.168.0.10 Accessed URL 10.20.1.20:/index.html
```

To enable FTP logging, enable fixup for FTP, set the logging level to 6 (informational), and look for message type of 303002. For example:

```
%PIX-6-303002:  192.168.0.10 Retrieved 10.20.1.20:file1.bin
%PIX-6-303002:  192.168.0.10 Stored 10.20.1.20:file2.bin
```

Logging Facility

Each syslog message has a facility number, which can be thought of as *where* the message should be logged. Twenty-four different facilities are available (refer to RFC 3164 for more information), with numerical codes ranging from 0 to 23. The eight facilities commonly used for syslog are local0 through local7. You can think of facilities as pipes leading to the syslogd process. The syslogd process files or places the messages into the correct log file based on the facility or inbound pipe. On the PIX firewall, facility configuration is optional. If used, the facility must be specified using its numerical code:

```
logging facility <facility_code>
```

Table 6.2 shows the facility names associated with each of the numerical codes.

Table 6.2 Facility Numerical Codes and Names

Numerical Code	Name
16	local0
17	local1
18	local2
19	local3
20	local4
21	local5
22	local6
23	local7

The default setting for facility configuration on a Cisco PIX is local4 (20). By changing the facility number, you can direct the syslog messages from different Cisco PIX firewalls (or even different types and models of devices) to different files. For example, on a Linux/UNIX machine, the /etc/syslog.conf file is configured with this:

```
# PIX Firewall syslog messages
local7.*      /var/log/pix/pix1
```

You can configure the PIX firewall to send syslog messages to the local7 log file (/var/log/pix/pix1) using the following command:

```
PIX1(config)# logging facility 23
```

Now the PIX will send syslog messages to facility local7 on the Linux server. Any syslog message arriving at the Linux syslogd process for facility local7 will be stored in the /var/log/pix/pix1 log file, whereas any syslog message for local4 (20) will continue to go to the default message log file.

Disabling Specific Syslog Messages

At times, you'll want to disable certain syslog messages. For example, let's say that you are logging all information while troubleshooting a connection and are constantly sending PING packets. You might want to disable any syslog message referencing ICMP to help diminish the flood of ICMP messages. In another example, say that someone has launched an attack against your PIX firewall and although you want a few of the messages, too many would fill up your disk space. Cisco provides a very useful document with a complete list of all syslog messages and their ID numbers. You can find it at www.cisco.com/univercd/cc/td/doc/product/iaabu/pix/pix_62/syslog/pixemsgs.htm. This document is not just for creating a list of disabled messages; it is also an excellent troubleshooting aid. Along with each syslog message are suggestions for what to do if the message is an error. For example, here is a complete syslog message explanation and a recommendation:

```
%PIX-1-103001: (Primary) No response from other firewall (reason code =
    code).
```

- **Explanation** This is a failover message. This message is logged if the primary unit is unable to communicate with the secondary unit over the failover cable. *(Primary)* can also be listed as *(Secondary)* for the secondary unit.
- **Action** Verify that the secondary unit has the exact same hardware, software version level, and configuration as the primary unit.

You can see that the document is very detailed. This document covers messages 100001 to 709007. There is one syslog message that you cannot disable:

```
%PIX-6-199002: PIX startup completed. Beginning operation.
```

In order to disable any other syslog message, use the following command:

```
no logging message <message_number>
```

The *message_number* parameter specifies the unique numeric message ID of each syslog message. For example:

```
PIX1(config)# no logging message 303002
```

In order to see which messages are disabled, use the *show logging disabled* command. For example:

```
PIX1# show logging disabled
no logging message 303002
```

In order to clear the disabled message so that it will be logged again, use the following command:

```
logging message <message_number>
```

The *message_number* parameter specifies the unique numeric ID of the disabled message. To re-enable all disabled messages, use the following command:

```
PIX1(config)# clear logging disabled
```

Configuring Remote Access

The ability to manage a Cisco PIX remotely is one of the blessings of remote management. You can always manage the PIX using the console port, but this requires you to be physically present at the PIX with a console connection. This solution is not very practical in today's enterprise networks. Fortunately, we have the option of using some type of remote access to manage the PIX. The tools we can use to remotely manage the PIX are Telnet, SSH, SNMP, or Cisco PDM. All these remote management methods have their place in the large picture of system management, but some fit better in certain situations than others. The goal of this section is to explain the various methods of remote management and show you the differences between them so you can make an educated judgment about which method to use in your specific situation.

NOTE

Terminal servers can enable remote access to the PIX firewall through the console port.

Two styles of remote access are available. The first and most commonly used is the *command-line interface,* or *CLI.* The CLI provides a very fast and low-overhead method of management. It also provides the ability to "cut and paste" configurations. The downside is that you need to know the commands and their

structures. Cisco PDM provides a more friendly method of managing the PIX remotely by providing a Windows-like GUI interface. You just point and click your way to configuring and monitoring the PIX firewall. The tradeoff is that PDM has a higher overhead requirement than the basic CLI. If you have a fat network pipe, such as a LAN connection, PDM makes good sense, but over a slow dialup connection, the lower overhead of the CLI makes it the preferred method of management. The CLI is remotely accessible through Secure Shell and Telnet.

Secure Shell

Secure Shell (SSH) is way to secure TCP/IP communication sessions by encrypting the data. With the encryption, neither the passwords nor the data are sent in clear text. SSH is not limited to the PIX but is used in a variety of ways, such as X.11 connections. SSH is intended to be a replacement for rlogin, rsh, and rcp, which are insecure protocols. For this discussion, we use SSH as the preferred method to connect to a Cisco PIX firewall instead of using the traditional Telnet method.

NOTE

The Cisco PIX firewall provides only the server component of SSH. A PIX cannot be an SSH client to another SSH server.

Whether you use Windows, UNIX, Linux, or Mac OS as your operating system, many SSH clients are available. For Windows, one of the most popular free clients is Tera Term with SSH extensions. For UNIX and Linux there is OpenSSH, and for the Mac there is NiftyTelnet. The examples that follow use Tera Term on a Windows platform.

NOTE

The PIX firewall only supports SSH version 1, not SSH version 2.

Enabling SSH Access

In order for the PIX to accept SSH connections, you must first enable SSH. Before you can use SSH, you need to generate an RSA key set. This RSA key is sent to the SSH server by the client to encrypt the session key. Do the following:

1. To generate the RSA key, the first step is to assign a hostname and a domain name to the PIX:

```
PIX1(config)# hostname PIX1
PIX1(config)# domain-name SecureCorp.com
```

2. Once you have completed assigning the hostname and the domain name, you need to generate the RSA key pair (one public key, and one private key) and save them to flash memory. The command to generate the pair of keys is:

```
ca generate rsa key <modulus>
```

   Cisco recommends 1024 bits for the **modulus**. This reflects RSA Security's own recommendations of using a key of 1024 bits for corporate use and 2048 bits for valuable keys. The larger the key, the longer it will take to generate the key and the longer it will take to crack it. The actual command for this example is as follows:

```
PIX1(config)# ca generate rsa key 2048
For <key_modulus_size> >= 1024, key generation could take up to
    several minutes. Please wait.
```

3. Once the generation process has completed, you can view the new RSA public key by entering the following command:

```
PIX1(config)# show ca mypubkey rsa
% Key pair was generated at: 13:13:04 UTC Aug 1 2002
Key name: PIX1.SecureCorp.com
 Usage: General Purpose Key
 Key Data:
  30820122 300d0609 2a864886 f70d0101 01050003 82010f00 3082010a
    02820101
  00b92dfe ac9a3fd1 f3c0bfd7 6920b498 b2722dbe d9aa8d4c f0bf0c0c
    a5bf1d3f
```

```
<<  output omitted  >>
% Key pair was generated at: 13:47:47 UTC Aug 10 2002
Key name: PIX1.SecureCorp.com.server
 Usage: Encryption Key
 Key Data:
  307c300d 06092a86 4886f70d 01010105 00036b00 30680261 00c150ba
    b244378c
<<  output omitted  >>
```

NOTE

If an RSA key is already saved on the PIX, you will be asked to remove the existing key. This is easily accomplished with the *ca zeroize rsa* command. This command clears the existing RSA key and allows you to generate a new RSA key set.

4. With the RSA key pair generated, you need to save it to flash using this command:

```
PIX501(config)# ca save all
```

5. Now you can configure the PIX for the allowed hosts or subnets that can be SSH clients to the firewall. You also can set the SSH inactivity timeout at this point. The format to allow SSH connections is:

```
ssh <ip_address> [<netmask>] [<interface>]
```

   If *netmask* is not specified, it is assumed to be 255.255.255.255; if *interface* is not specified, it is assumed to be the inside interface. In the following example, *ip_address* is 192.168.50.0 and *netmask* is 255.255.255.0. This allows the entire 192.168.50.0/24 subnet range SSH access to the PIX. The *interface* parameter specifies the name of the interface on which this subnet resides. In this case, it is the inside interface.

```
PIX1(config)# ssh 192.168.50.0 255.255.255.0 inside
```

6. By default, the PIX will disconnect an SSH session after 5 minutes of inactivity. We can set the inactivity timeout between 1 and 60 minutes. To set the inactivity timeout to 10 minutes, use the following command:

```
PIX1(config)# ssh timeout 10
```

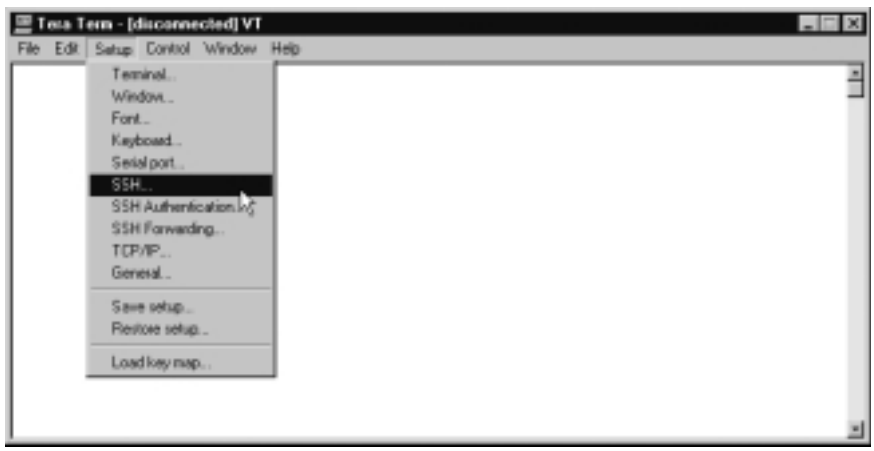
7. Finally, we need to save the changes to flash:

```
PIX1# write memory
```

To verify the SSH configuration, use the *show ssh* command in Enable mode.

To access the PIX firewall, you need to configure an SSH client. In this example, we use a popular Windows SSH client, Tera Term. Tera Term and SSH Extensions can be downloaded from www.zip.com.au/~roca/ttssh.html. First, install Tera Term. When Tera Term is installed, follow the directions in the Readme file to install the SSH extensions into the root directory for Tera Term. Once the SSH extensions are installed, you need to specify an ssh_known_hosts file. Figure 6.4 shows where to find the SSH setup menu in Tera Term.

Figure 6.4 Configuring SSH in Tera Term

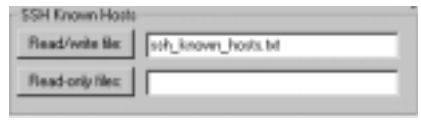


When you click the **SSH** menu item, you will see a dialog box (see Figure 6.5) that has two items that need to be configured. The first item is the preferred cipher order. In this configuration, DES is configured to be first since this particular PIX firewall does not have 3DES enabled.

Figure 6.5 Selection of Ciphers in Tera Term SSH

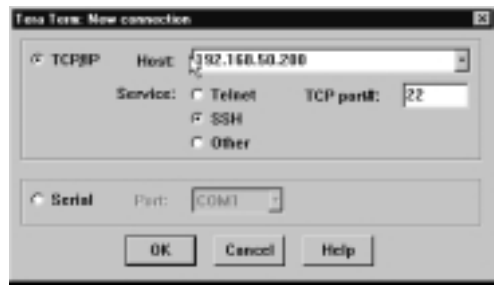
The second item to configure is the ssh_known_hosts file, as shown in Figure 6.6. This can be a blank text file to which Tera Term will add known hosts and keys.

Figure 6.6 Configuring the ssh_known_hosts.txt File



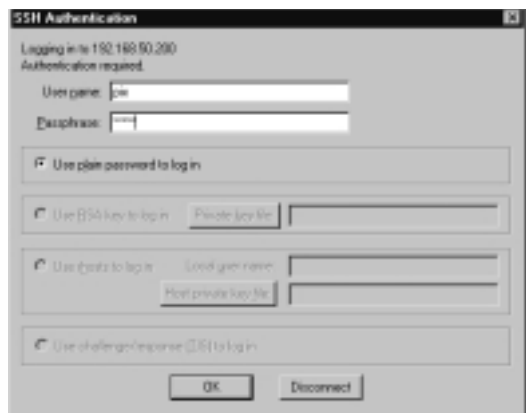
When you start Tera Term, a dialog box opens (see Figure 6.7). You need to type in the IP address of the PIX firewall and choose the type of service by clicking a radio button. The default service is Telnet, so make sure that you select **SSH** and then click **OK**.

Figure 6.7 Configuring a New Connection in Tera Term



After a moment, you will be presented with the next screen, shown in Figure 6.8.

Figure 6.8 SSH Authentication



The default username for a Cisco PIX SSH connection that is *not* using AAA for authentication is *pix*. The passphrase is the password that is used for Telnet. Once the username and passphrase are authenticated, your SSH session will start. This authentication can take a few moments, so be prepared to wait a bit. Figure 6.9 shows the completed SSH connection to the Cisco PIX. A small icon in the upper-left corner of Tera Term shows that you have an SSH connection.

Figure 6.9 Verifying the SSH Connection

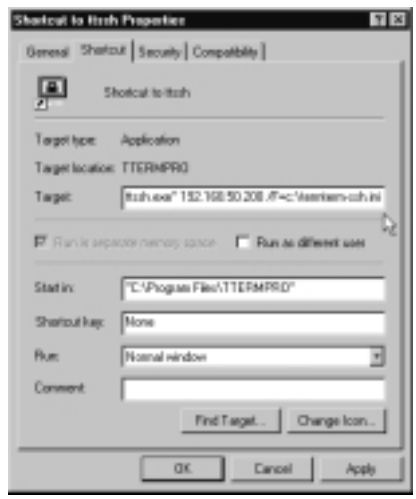
To configure Tera Term to automatically use SSH and a certain IP address, first configure Tera Term with the correct encryption, screen colors, and other settings and then save the setup with a name of your choice by clicking **Setup | Save Setup**, as shown in Figure 6.10.

Figure 6.10 Saving the Tera Term Configuration

Once that process is completed, create a shortcut to the Tera Term application. Right-click the shortcut that you just made and choose **Properties** in the dialog box. The Target entry line will show where Tera Term is located and any

parameters with which Tera Term will start. Add two items to the parameters. The first one is the IP address of the PIX firewall and the second is the */F* switch, which will allow you to specify the ini file that you saved in the prior step. You need to specify the path for the */F* switch, as shown in Figure 6.11. Click **OK** and you are set. The next time you start Tera Term with this shortcut, it will load the saved .ini file and automatically connect to the target host.

Figure 6.11 Editing the Tera Term Shortcut



Troubleshooting SSH

At times you will need to troubleshoot the reason that the SSH connection is failing. In this case, use the *debug ssh* command on the PIX. The debug output on PIX is relatively easy to understand and can be read easily without much trouble. Figure 6.12 shows the output of the *debug ssh* command for a successful SSH connection.

Figure 6.12 An Example of a Successful SSH Connection

```
152: SSH: Device opened successfully.
153: SSH: host key initialized
154: SSH0: SSH client: IP = '192.168.50.7'  interface # = 1
155: SSH0: starting SSH control process
156: SSH0: Exchanging versions - SSH-1.5-Cisco-1.25

157: SSH0: send SSH message: outdata is NULL
```

```
158: SSH0: receive SSH message: 83 (83)
159: SSH0: client version is - SSH-1.5-TTSSH/1.5.4 Win32
160: SSH0: begin server key generation
161: SSH0: complete server key generation, elapsed time = 4170 ms
162: SSH0: declare what cipher(s) we support: 0x00  0x00  0x00  0x04
163: SSH0: send SSH message: SSH_SMSG_PUBLIC_KEY (2)
164: SSH0: SSH_SMSG_PUBLIC_KEY message sent
165: SSH0: receive SSH message: SSH_CMSG_SESSION_KEY (3)
166: SSH0: SSH_CMSG_SESSION_KEY message received - msg type 0x03, length
     272
167: SSH0: client requests  DES cipher: 2
168: SSH0: send SSH message: SSH_SMSG_SUCCESS (14)
169: SSH0: keys exchanged and encryption on
170: SSH0: receive SSH message: SSH_CMSG_USER (4)
171: SSH0: authentication request for userid PIX
172: SSH(PIX): user authen method is 'no AAA', aaa server group ID = 0
173: SSH0: send SSH message: SSH_SMSG_FAILURE (15)
174: SSH0: receive SSH message: SSH_CMSG_AUTH_PASSWORD (9)
175: SSH0: send SSH message: SSH_SMSG_SUCCESS (14)
176: SSH0: authentication successful for PIX
177: SSH0: receive SSH message: SSH_CMSG_REQUEST_PTY (10)
178: SSH0: send SSH message: SSH_SMSG_SUCCESS (14)
179: SSH0: receive SSH message: SSH_CMSG_EXEC_SHELL (12)
180: SSH0: starting exec shell
```

Figure 6.13 shows an example of an incorrect username. The Cisco PIX firewall will reject the login even if the password is correct.

Figure 6.13 An Example of an Incorrect User Name

```
184: SSH: Device opened successfully.
185: SSH: host key initialised
186: SSH0: SSH client: IP = '192.168.50.7'  interface # = 1
187: SSH0: starting SSH control process
188: SSH0: Exchanging versions - SSH-1.5-Cisco-1.25
189: SSH0: send SSH message: outdata is NULL
190: SSH0: receive SSH message: 83 (83)
191: SSH0: client version is - SSH-1.5-TTSSH/1.5.4 Win32
```

```
192: SSH0: begin server key generation
193: SSH0: complete server key generation, elapsed time = 7090 ms
194: SSH0: declare what cipher(s) we support: 0x00  0x00  0x00  0x04
195: SSH0: send SSH message: SSH_SMSG_PUBLIC_KEY (2)
196: SSH0: SSH_SMSG_PUBLIC_KEY message sent
197: SSH0: receive SSH message: SSH_CMSG_SESSION_KEY (3)
198: SSH0: SSH_CMSG_SESSION_KEY message received - msg type 0x03, length
    272
199: SSH0: client requests  DES cipher: 2
200: SSH0: send SSH message: SSH_SMSG_SUCCESS (14)
201: SSH0: keys exchanged and encryption on
202: SSH0: receive SSH message: SSH_CMSG_USER (4)
203: SSH0: authentication request for userid badname
204: SSH(badname): user authen method is 'no AAA', aaa server group ID = 0
205: SSH0: invalid userid badname
206: SSH0: send SSH message: SSH_SMSG_FAILURE (15)
207: SSH0: receive SSH message: SSH_CMSG_AUTH_PASSWORD (9)
208: SSH0: send SSH message: SSH_SMSG_FAILURE (15)
209: SSH0: receive SSH message: SSH_MSG_DISCONNECT (1)
210: SSH0: authentication failed for badname
211: SSH0: Session disconnected by SSH server - error 0x36 "Reset
    by client"
```

To see how many SSH sessions are on the PIX, use the following command:

```
show ssh sessions [<ip_address>]
```

The optional *ip_address* parameter allows you to check for SSH sessions from a particular IP address. An example of the results of using this command follows:

```
PIX1# show ssh sessions

Session ID       Client IP        Version Encryption       State    Username
1                192.168.50.8      1.5      DES             6        pix
```

To disconnect a specific SSH session, use this command:

```
ssh disconnect <session_id>
```

For example:

```
PIX1(config)# ssh disconnect 0
```

The *session_id* parameter specifies the number associated with the SSH session that is shown by using the *show ssh sessions* command.

To remove all SSH configuration statements from the Cisco PIX, use this command:

```
PIX1(config)# clear ssh
```

Telnet

Telnet is one of the simplest ways to connect to a network device. Telnet is character based and sends each character in clear text across the network. This means that any username and password are available to anyone who has a way to capture packets. As you might imagine, this vulnerability leads to a very large security risk in managing the PIX across a WAN link, the Internet, or even on a LAN. This is an excellent reason to configure and use SSH instead of Telnet.

NOTE

The Cisco PIX firewall can only be a Telnet server and not a Telnet client. This is unlike Cisco routers and switches, on which you can Telnet from one system to the next.

To configure Telnet on the PIX firewall, use the following command.

```
telnet <ip_address> [<netmask>] [<interface>]
```

The *ip_address* parameter can be a single IP host or an entire range. The *netmask* parameter is a subnet mask associated with the IP address or range. The *interface* parameter specifies the interface name you want to enable Telnet on. In the following example, we have configured the entire 192.168.50.0/24 network on the inside interface to be allowed to Telnet to the PIX:

```
PIX1(config)# telnet 192.168.50.0 255.255.255.0 inside
```

There are some restrictions on Telnet depending on the interface you use. The inside interface allows Telnet without encryption, whereas encryption (through VPN) is needed for the outside interface. You can set the idle timeout value for the Telnet session. The timeout value is specified in minutes and must be a value from 1 to 60. The default timeout is 5 minutes. In this example, we configure the timeout to 15 minutes:

```
PIX1(config)# telnet timeout 15
```

The *show telnet* command shows the current list of IP addresses and their interfaces that are authorized to access the PIX via Telnet. For example:

```
PIX1# show telnet
192.168.50.0 255.255.255.0 inside
```

The *clear telnet* or *no telnet* commands remove the Telnet privilege from an authorized IP address. The format for this command is:

```
clear telnet [<ip_address> [<netmask>] [<interface>]
```

The *ip_address* parameter is the subnet or host IP that you want to clear. The *netmask* parameter is a subnet mask associated with the IP address or range. The *interface* parameter is the name of the interface that had this Telnet host or subnet IP enabled. For example:

```
PIX1(config)# clear telnet 192.168.50.0 255.255.255.0 inside
```

If no parameters are specified, the *clear telnet* command will remove access for all hosts.

The *who* command shows you which IP addresses currently have Telnet sessions open to the PIX. This example shows two Telnet sessions, one from 192.168.50.3 and the other from 192.168.50.8:

```
PIX1# who
        0: 192.168.50.3
        1: 192.168.50.8
```

The *kill <telnet_id>* command terminates an active Telnet session. No warning is given the user when the session is dropped. The *telnet_id* parameter specifies the session number that is shown when you use the *who* command. For example:

```
PIX1# kill 0
```

Restrictions

Before PIX software version 5.0, you could only Telnet to the PIX from the inside interface. With PIX OS 5.0 and later versions, you can Telnet to any interface, but the PIX firewall requires all Telnet traffic to the outside interface to be protected using encryption. It is possible to use access lists and a static route to pass a Telnet session through the Cisco PIX from the outside interface to a Telnet

server on the inside and then use the Telnet server to Telnet back to the inside interface. However, it is much easier to use SSH to open a CLI session to the outside interface of the firewall. This meets the Cisco PIX requirement of using an encrypted connection for the Telnet session. You can use Telnet to an outside interface over an encrypted VPN session.

HTTP Via the PIX Device Manager

Cisco PIX Device Manager (PDM) is a Web-based Java applet that gives you roughly 98 percent of what can be configured at the command line. The full scope of PDM is covered in Chapter 9, but in this section, we give you a very quick look at the possibilities of what PDM can offer. Cisco PDM will give you a GUI interface that allows you to quickly configure and manage a single PIX firewall. PDM makes use of tabs, drop-down menus, and other GUI tools to provide an easy administration interface. PDM also offers graphs of firewall and traffic activity for viewing and printing.

Configuring Simple Network Management Protocol

Simple Network Management Protocol (SNMP) is one of the easiest ways to manage a network device and to retrieve information from it. Many readers will be familiar with SNMP on Cisco routers, but on the Cisco PIX, things are a bit different. SNMP on the Cisco PIX is read only.

Do not use a weak SNMP community string. You should never use the default of *public* as the SNMP string. This well-known and weak string defeats the purpose of trying to secure your PIX firewall. The string you choose should not be a dictionary-based word. For example, *UcanN0tGuEe$$ME* would be a very difficult community string to guess, and most dictionary attacks would fail against it.

There are three versions of SNMP. Here we concentrate on version 1 because that is the version the PIX firewall supports. Various SNMP managers are available to manage the PIX firewall using SNMP. We have listed a few of them here:

- HP OpenView
- SolarWinds
- CiscoWorks
- Castle Rock SNMPc
- The Multi Router Traffic Grapher (MRTG)

The one SNMP application that deserves special mention is the Multi Router Traffic Grapher (MRTG). Strictly speaking, MRTG is not an SNMP manager application but a graphing application that uses SNMP to gather data and generate graphs. MRTG generates graphs based on polled SNMP values. These graphs can be then inserted into documents, Web pages, or e-mail. MRTG is free for download and is available at www.mrtg.org. MRTG works well with the PIX firewall. An example of using MRTG with the PIX firewall can be found at www.somix.com/software/mrtg. This Web site provides a script for monitoring the number of connections on a PIX firewall.

In order to make good use of SNMP to monitor the PIX firewall, you need to download the Cisco PIX Management Information Bases (MIBs). These MIBs can be found at www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml. Once you have downloaded the MIBs, you need to compile them in your SNMP manager before you can use them to manage the Cisco PIX beyond some simple OIDs.

There are two ways to get SNMP information from the PIX firewall. The first is to query the PIX using SNMP. The host will send a query to the PIX (also known as *polling* it for information) and receive a response. The second way is to have the PIX send "traps" to the SNMP management station. The traps sent are not the same as polled OIDs. A *trap* is a message that the PIX sends based on an event that has occurred, such as a link going up or down or a syslog event. Polling can be used to retrieve information or values that can be displayed by the SNMP management station in the form of gauges, bar charts, or another format. Polling can also retrieve system information about the PIX, such as the software version, interface statistics, and CPU utilization.

Configuring System Identification

Basic SNMP identification is easily configured on the PIX firewall and is accomplished using the following configuration mode commands:

```
snmp-server location <word>
snmp-server contact <word>
```

Both of these commands are optional. The *word* parameter in both commands can be any string up to 127 characters. The location can describe a building, closet, rack location, or any other standard that you use on your network. The contact can be contact person or company that is responsible for administering the PIX. Verify SNMP configuration using the *show snmp* Enable mode command.

Configuring Polling

One way to gather performance and statistical data from a Cisco PIX firewall is to use SNMP polling. Configuring SNMP polling on the Cisco PIX allows an SNMP management station to retrieve data using PIX SNMP OIDs. To configure polling, first make sure that an SNMP community is set using the following Configuration mode command:

```
snmp-server community <word>
```

The *word* parameter specifies the SNMP community (the password). You should not use easily guessed words or the commonly used PUBLIC string. There are many free dictionary-based SNMP community string crackers, so for a secure community string, do not use a plain dictionary-based string. This parameter is required for SNMP to function correctly, is case sensitive, and is limited to 32 characters. For polling to work, the PIX firewall must be configured with the IP address of the polling station. This is accomplished using the following command:

```
snmp-server host [<interface>] <ip_address> poll
```

The *ip_address* parameter is the IP address of the SNMP management station. The *interface* parameter specifies the interface where the management station is located. If the interface is not specified, it is assumed to be the inside interface. The *poll* parameter specifies that the management station will query the PIX. You may specify multiple polling station IP addresses by typing multiple *snmp-server host* commands.

Castle Rock SNMPc is an SNMP manager that can be found at www.castlerock.com/products/products.htm. In Figure 6.14, it is being used to poll the Cisco PIX firewall for system information.

In Figure 6.15, we are using the Castle Rock MIB browser to drill down into the ciscoFirewallMIB to look at the OID of 1.3.6.1.4.1.9.9.147.1.2.2.2.1.4, which gives the status of the connection count variable. This OID is one of the favorites to watch if the PIX does not have an unlimited license. Other favorite OIDs are shown in Table 6.3. To find all the OIDs for the PIX firewall, go to ftp://ftp.cisco.com/pub/mibs/oid/ and download the appropriate MIB.

Figure 6.14 Castle Rock SNMPc Manager Polling a PIX Firewall

Figure 6.15 Browsing the PIX MIB

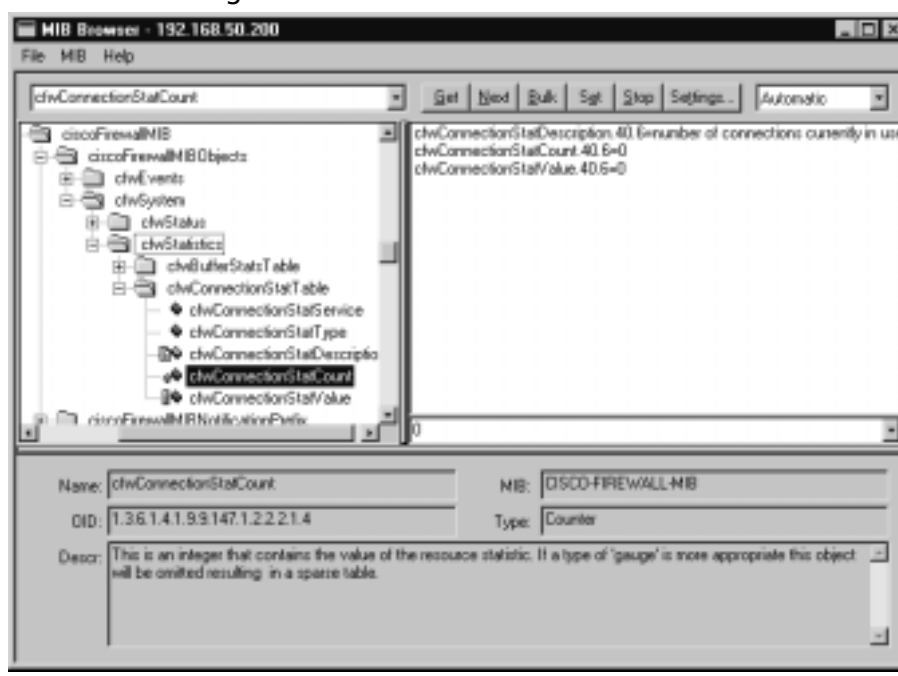


Table 6.3 Useful Cisco PIX OIDs

Description	OID
System description	1.3.6.1.2.1.1.1.0
System uptime	1.3.6.1.2.1.1.3.0
Memory used	1.3.6.1.4.1.9.9.48.1.1.1.5.1
Memory free	1.3.6.1.4.1.9.9.48.1.1.1.6.1
Failover status	1.3.6.1.4.1.9.9.147.1.2.1.1.1.4.7
Current connections in use	1.3.6.1.4.1.9.9.147.1.2.2.2.1.5.40.6
Most connections in use	1.3.6.1.4.1.9.9.147.1.2.2.2.1.5.40.7
CPU utilization (5 second)	1.3.6.1.4.1.9.9.109.1.1.1.1.3.1
CPU utilization (1 minute)	1.3.6.1.4.1.9.9.109.1.1.1.1.4.1
CPU utilization (5 minute)	1.3.6.1.4.1.9.9.109.1.1.1.1.5.1

Configuring Traps

In simple terms, SNMP traps are messages that are triggered by an event such as an interface going down. The SNMP traps are sent on UDP port 162 and are not encrypted. To configure and use SNMP traps, follow these steps:

1. Configure the SNMP community.

```
PIX1(config)# snmp-server community Il0v3CiSCo
```

2. Configure the SNMP host that will receive the traps. The syntax is similar to configuring a host for polling, except the *trap* keyword is used instead of *poll*:

```
PIX1(config)# snmp-server host inside 192.168.50.8 trap
```

NOTE

If you configure an SNMP host without using the *poll* or *trap* keywords, the SNMP host will be used for both functions.

3. Enable SNMP traps:

```
PIX1(config)# snmp-server enable traps
```

4. Set the logging level for SNMP traps using the *logging history* command. For example:

```
PIX1(config)# logging history errors
```

5. Start sending traps to the SNMP management station using the *logging on* command:

```
PIX1(config)# logging on
```

To stop SNMP traps, use the *no snmp-server enable traps* command.

Configuring System Date and Time

An accurate system clock is one of the most overlooked and yet vital system management requirements. Many pieces of system management and security depend on an accurate time and date mechanism.

You might ask why configuring the clock and time zone is so important. In managing the PIX firewall, the clock and time zone allow you to build an accurate timeline of what has happened in the log files. For example, if you need to build a legal case and use the log files from the PIX, the courts will expect the logs to be in the Coordinated Universal Time (UTC) format and they must be consistent across all the devices. This log file timestamp consistency provides the one constant reference point across the network. Without this consistency across all the log files, it becomes difficult, if not impossible, to rebuild an incident's timeline.

When we speak of UTC, we are referring to the older standard, Greenwich Mean Time (GMT). Cisco's implementation of the Public Key Infrastructure (PKI) also uses the clock to verify that the certificate revocation list (CRL) has not expired. If the clock is not correct, the certificate authority (CA) may reject or allow a digital certificate based on the incorrect clock timestamp.

As vital as an accurate clock is, many times it is overlooked or viewed as "too much trouble" to configure. The good news is that the PIX firewall is easily configured for an accurate system time and date and provides an easy way to keep the clock consistent and accurate across time zones using NTP.

In this section, you will learn how easy it is to configure the PIX firewall for an accurate clock and how NTP can be used for ease of managing the clock. You will learn how to configure the PIX for daylight savings time. You will also learn that multiple PIX firewalls can be set from a central server and how to configure NTP on the PIX to use a central server securely.

Setting and Verifying the Clock and Time Zone

New enhancements to the PIX firewall allow you adjust both the viewed time zone information and account for daylight savings time. These enhancements allow you to view the clock information in a readily understandable time format without having to convert the internal UTC into your local time.

NOTE

The benefit of using UTC is that no matter where you are located, the base time is always the same. You either add or subtract a number of hours from UTC to get the local time. This provides a consistent time across an enterprise network that spans various time zones.

There are three suggestions for configuring the PIX clocks across an enterprise network:

- Always display the "local" time zone for each device, based on where the device is located. This practice is useful when you have regional administrators for the firewalls and they frequently administer and monitor their local PIX firewalls.
- Set all devices internally to the UTC format for a standard clock across multiple time zones. We already discussed the benefits of using this method.
- Set all devices to display the local "headquarters" time zone. This is useful when you have a centralized IT staff that manages firewalls across the globe.

To check the time on a PIX firewall, type *show clock*. If you decide to use the local clock on the PIX, use this command to set the clock:

```
clock set <hh:mm:ss month day year>
```

The *hh:mm:ss* parameter is the normal hours:minutes:seconds in the 24-hour format. The month should be the first three characters of the month, then the day using numerals 1 to 31, and lastly the year, from 1993 to 2035. PIX version 6.2 supports daylight savings time (*summer-time*) and time zones.

The command format to set the *summer-time zone* parameter is as follows:

```
clock summer-time <zone> date <week weekday month hh:mm week weekday month
    hh:mm [offset]>
```

The *zone* parameter is the name of the time zone, such as PST. The other parameters are used to set the start and the end of summer time. If you want to make this a recurring event, change the command slightly:

```
clock summer-time <zone> recurring <week weekday month hh:mm week weekday
    month hh:mm [offset]>
```

The new parameter *recurring* will start and stop the *summer-time* adjustment each year at the same point. Here is an example:

```
PIX1# show clock
04:22:19.659 UTC Mon Oct 7 2002
PIX1# configure terminal
PIX1(config)# clock summer-time pst date 7 april 2002 00:00 27 october
    2002 00:00
PIX1(config)#
PIX1# show clock
05:23:02.890 pst Mon Oct 7 2002
PIX1# show clock detail
05:23:05.751 pst Mon Oct 7 2002
Time source is user configuration
Summer time starts 00:00:00 UTC Sun Apr 7 2002
Summer time ends 00:00:00 pst Sun Oct 27 2002
```

To set the time zone for the display only, use the following Configuration mode command:

```
clock timezone <zone> <hours> [<minutes>]
```

Keep in mind that *clock timezone* will only set the displayed time; the internal time is still kept in UTC format. The *zone* parameter is the name of the time zone. The *hours* parameter is the time offset from UTC. To disable the time zone, type in **no clock timezone**.

In order to clear the clock settings, you can use the *clear clock* command. You can see in the following example that the command cleared the *summer-time* settings:

```
PIX1# show clock detail
17:01:43.480 pst Fri Sep 20 2002
Time source is user configuration
Summer time starts 00:00:00 UTC Sun Apr 7 2002
Summer time ends 00:00:00 pst Sun Oct 27 2002
PIX1# configure terminal
```

```
PIX1(config)# clear clock
PIX1# show clock detail
16:02:36.301 UTC Fri Sep 20 2002
Time source is user configuration
```

Configuring and Verifying the Network Time Protocol

It is possible to just set the clock and time on a single PIX firewall, but trying to set an accurate time and date stamp on multiple Cisco PIX firewalls can be a serious and time-consuming management problem. A preferred solution is to use the Network Time Protocol (NTP). NTP uses servers as the master reference point, and the NTP client, in this case a PIX firewall, will use the NTP server to get accurate time. The NTP server gets its own time from a radio source or atomic clock. The NTP servers listen on UDP port 123 for requests. The Cisco PIX firewall queries an NTP server and updates its clock. Once NTP is configured on the all the PIX firewalls, all the log files will have consistent and accurate timestamps.

There are two strata, or classes, of NTP servers. Stratum 1 NTP servers are directly connected to the time source. Stratum 2 servers are the second level and consider Stratum 1 servers to be authoritative. Cisco supports only Stratum 2 servers.

You can get the time from public Stratum 2 servers on the Internet or you configure your own NTP server on the LAN or WAN. A quick search for public NTP servers on the Internet reveals many public NTP Stratum 2 servers that you can use. To enable the Cisco PIX Firewall NTP client, use the following command:

```
ntp server <ip_address> source <interface>
```

The *ip_address* parameter specifies the IP address of the NTP server from which you want the Cisco PIX to get its time. The *interface* parameter specifies the source interface on which the PIX firewall will find the NTP server. To remove an NTP server, use the following command:

```
no ntp server <ip_address>
```

The following example shows this command and how to check the configuration to make sure the PIX is talking with the timeserver correctly using the *show ntp status* and *show ntp association* commands:

```
PIX1(config)# ntp server 192.168.1.3 source inside
PIX1(config)# show ntp status

Clock is unsynchronized, stratum 16, no reference clock
nominal freq is 99.9967 Hz, actual freq is 99.9967 Hz, precision is 2**6
reference time is 00000000.00000000 (06:28:16.000 UTC Thu Feb 7 2036)
clock offset is -4.0684 msec, root delay is 0.00 msec
root dispersion is 0.00 msec, peer dispersion is 15875.02 msec
PIX1(config)# show ntp associations

      address          ref clock      st   when   poll reach   delay   offset
    disp
 "192.168.1.3       0.0.0.0           16      -     64     0      0.0     0.00
    16000.
 master (synced), # master (unsynced), + selected, - candidate, "
    configured
```

You can view the NTP configuration using the *show ntp* command in Enable mode. To delete the NTP configuration, all you need to do is enter the *clear ntp* command in Configuration mode. That's it; the NTP configuration will be completely cleared.

NTP Authentication

Given that we are dealing with a security device, we should always try to enable NTP authentication. One of the dangers of not using NTP authentication is that a clever hacker could reset the clock, which in turn would change the log file time-stamps and possibly help cover up the signs of the security breach. Another hack would be to get around time-based security by resetting system clocks, sending packets to the Cisco PIX with forged information. Setting up NTP authentication on the PIX is simple. The authentication uses trusted keys to provide the authentication between the NTP server and the client. In order to authenticate, the authentication key on the PIX must match the authentication key on the server, which is a string that can be up to 32 characters, including spaces.

NTP authentication is disabled by default on the PIX. To configure NTP authentication, first start with enabling NTP authentication using the following command:

```
ntp authenticate
```

Now you need to define the authentication key. The only choice of encryption is MD5:

```
ntp authentication-key <number> md5 <value>
```

The *number* parameter is a value from 1 to 4294967295 that uniquely identifies the key. The *value* parameter is an arbitrary string of 32 characters, including all printable characters and spaces.

Now we define the trusted key that will be sent in the NTP packets:

```
ntp trusted-key <key_number>
```

The *key_number* parameter must be a number from 1 to 4294967295. The last step is to configure the server association, which lets the Cisco PIX firewall synchronize to the other server. Use the following command:

```
ntp server <ip_address> key <number> source <if_name> [prefer]
```

NOTE

The Cisco PIX will not let other time servers synchronize to itself. NTP synchronization is a one-way street as far as the PIX firewall is concerned. It is a client and only a client.

The *ip_address* specifies the IP address of the server to which you want the Cisco PIX to authenticate. The next piece, *key*, is the number of the shared key that you used when you configured the trusted-key command. The last part, *interface*, is the interface that will send the NTP packets to the server. The optional *prefer* keyword will have the Cisco PIX go to this server first to set the time.

Here is an example of configuring NTP authentication:

```
PIX1(config)# ntp authenticate
PIX1(config)# ntp authentication-key 10 md5 ciscoisgreat
PIX1(config)# ntp trusted-key 10
PIX1(config)# ntp server 192.168.50.3 key 10 source inside
PIX1(config)# show ntp
ntp authentication-key 10 md5 ********
ntp authenticate
ntp trusted-key 10
ntp server 192.168.50.3 key 10 source inside
```

Summary

In this chapter, we have seen that network management, although appearing simple on the surface, can be quite complex. To effectively manage the Cisco PIX, you need to be aware of not just the PIX but also networkwide issues.

When configuring the PIX for logging, you can make a choice from a variety of logging paths, such as buffered logging, console, Telnet/SSH sessions, syslog servers, or SNMP. With each of the logging paths, you can select message severity levels ranging from Level 1 (alert) to Level 7 (debug) based on your needs. Aside from selecting the severity level, you can choose from several facility levels to direct the flow of the syslog messaging. The default facility level is local4 (20), but you can use other facility levels to redirect syslog messages from different sources to a syslog server destination of your choice. This system provides a method to store various sources of syslog messages in their own files on the syslog server.

You can specify that all syslog messages should be logged or you can filter out certain messages so they will not be sent. This functionality is very useful in troubleshooting a network issue where you might be in Debug mode and the normal message flow would be overwhelming to work with.

The Cisco PIX firewall can be managed using a console port, but most of the time the PIX will be managed by remote access. Two popular choices of protocol for remote access are Telnet and SSH. Telnet has been around for a long time and is used on a variety of network devices, but it is an insecure protocol and sends the information in clear text across the network. SSH, on the other hand, encrypts the session so that information such as passwords is not sent in clear text. SSH also provides a way to be able to log into the outside interface of the Cisco PIX, unlike Telnet, which is not permitted to directly log into the outside interface without an encrypted connection. The Cisco PIX firewall can only act as a server for SSH and Telnet services, not a client.

An alternative method of accessing the PIX firewall remotely for system management is the Cisco PDM utility. PDM is a Java application that allows the management of the Cisco PIX using a Web browser. PDM has good reporting functionality to build graphs showing various performance statistics, attack reports, and traffic activity.

The Cisco PIX supports read-only SNMP reporting and can either send traps to a host or be polled for information.

The Cisco PIX firewall has a wealth of system time and date functionality. This functionality goes from the basic time and date stamp to automatically

adjusting for daylight savings time. The Cisco PIX clock can be set locally or NTP can be used to set the time from a central timeserver. The PIX uses the UTC time format but can be configured to display the time in a time-zone format such as PST. The PIX can use NTP authentication to keep the link to the timeserver secure from unauthorized adjustment of the system time. This provides a level of security for using digital certificates.

Solutions Fast Track

Configuring Logging

- ☑ All logging on the PIX is disabled by default. Once you have configured logging, do not forget to turn it on using the *logging on* command.
- ☑ You can view the log messages from the console, through Telnet/SSH sessions, using syslog servers, using SNMP, or using Cisco PDM. You can also use a combination of these methods.
- ☑ Syslog functionality on the PIX provides a way to send logging messages to a remote server using either UDP or TCP connections.
- ☑ Eight levels of message severity are available, but the PIX only uses seven. The PIX does not use Level 0.
- ☑ Caution must be exercised when enabling logging and setting the logging level, because the number of logging messages can easily overwhelm a production PIX.

Configuring Remote Access

- ☑ Telnet is an insecure protocol and sends information across the network in clear text. Therefore, it is recommended that SSH be used for remote management of the PIX.
- ☑ You cannot Telnet directly to the outside interface unless the connection is encrypted.
- ☑ In order for SSH to function, DES or 3DES must be enabled on the PIX.

- ☑ PDM provides a GUI method to easily configure, manage, and view statistics on the Cisco PIX firewall.
- ☑ In order for SSH to function, you must first generate RSA keys using the *ca generate rsa key <modulus>* command and save them using the *ca save all* command.

Configuring Simple Network Management Protocol

- ☑ SNMP on the Cisco PIX firewall is *read only*.
- ☑ The community string is the password to the SNMP information and should not be an easily guessed or easily cracked string. Remember that the community string is case sensitive.
- ☑ The PIX firewall can be queried ("polled") from an SNMP device. It also has the ability to send SNMP traps.
- ☑ To fully utilize SNMP management on the PIX, you need to get the PIX MIBs from Cisco and compile them with your SNMP management application.

Configuring System Date and Time

- ☑ The PIX internal clock uses UTC time, but you can set the display to be your normal time zone.
- ☑ NTP should be used to automate and provide a single clock source for the enterprise network. This provides a consistent and accurate time across all devices.
- ☑ NTP is insecure and should be configured using encryption for maximum protection.
- ☑ The Cisco PIX firewall will not act as a NTP server; it will only be a NTP client.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: I see an error message such as "201008: The PIX is disallowing new connections." Now my PIX will not pass any inbound or outbound traffic. What has happened?

A: Your Cisco PIX is configured to use TCP syslog, and something has happened to break the TCP connection between the PIX and the syslog server. It could be that the service has stopped or even that the allocated message storage is full. Either correct the problem or use the UDP syslog service.

Q: My SSH session constantly fails to connect with the PIX. Why is this happening?

A: The most common reason is that the RSA key was generated but not saved. Regenerate the key, and be sure to use the *ca save all* command.

Q: I have configured syslog on my PIX firewall, and the syslog server has been configured. However, no messages are being logged. What is wrong?

A: On both the PIX and syslog server, the protocol and port number need to be the same. In addition, make sure that the facility is the same. The default is local4 (20), so if you have changed this setting, it needs to be changed on both sides.

Q: When I poll my PIX using SNMP, the throughput performance of the PIX degrades. What can I do?

A: If too many SNMP OIDs are being polled at once or too often, the PIX processor can be overloaded to the point where throughput will suffer. Check your SNMP management station and see which variables are being polled and how often. A second SNMP issue can be that the severity level of the traps is set too high and too many traps are being sent to the SNMP management station. A classic example is that the severity level has been set to

debugging to troubleshoot a problem and then forgotten about until a performance degradation is noticed.

Q: When I use PDM to view graphs under the Monitoring tab, the time is incorrect.

A: PDM assumes that the PIX clock is set to UTC format. PDM then adds or subtracts the difference between the UTC and your time zone. The resulting time is what is used on the graphs. This situation is easily corrected using the *clock* command.

Q: I have configured my PIX firewall to use authenticated NTP, but I cannot connect to the timeserver. Why not?

A: Encrypted NTP requires the use of authentication keys. These keys *must* match on the PIX and the NTP server. If they do not match, the PIX will not be able to connect to the NTP server and receive updates.

Chapter 7

Configuring Virtual Private Networking

Solutions in this chapter:

- IPsec Concepts
- Configuring Site-to-Site IPsec using IKE
- Configuring Site-to-Site IPsec Without IKE (Manual IPsec)
- Configuring Point-to-Point Tunneling Protocol
- Configuring Layer 2 Tunneling Protocol with IPsec
- Configuring Support for the Cisco Software VPN Client

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

Virtual private network (VPN) technology provides a channel for secure communications between internal networks over a public network (such as the Internet, for example) while providing features such as confidentiality and authentication. VPNs are commonly used to connect branch offices, mobile users, and business partners. The ability to connect private networks or hosts by securely tunneling through a public network infrastructure is very appealing. VPNs over the Internet provide solutions to various business problems, including economical connectivity between offices (using site-to-site VPNs) and the ability to provision connections quickly (simply by installing VPN hardware on an existing Internet connection instead of having to wait for a dedicated leased line or Frame Relay PVC to be installed). Remote access VPNs, on the other hand, provide connectivity for mobile workers or telecommuters, allowing them to dial into any ISP or use high-speed broadband connectivity at home or at a hotel to gain access to the corporate network.

The PIX firewall supports both site-to-site and remote access VPNs using various protocols: IPsec, L2TP, and PPTP. On the technical side, VPNs can be very complicated, and a single connection might be implemented using a combination of many protocols that work together to provide tunneling, encryption, authentication, access control, and auditing.

In this chapter, you will learn how to configure VPN on the PIX firewall. We will configure site-to-site VPNs (also known as *office-to-office VPNs*) using IPsec and IKE with pre-shared keys and digital certificates. You will also learn about manual IPsec and how to configure PPTP and L2TP tunneling on the PIX firewall. Finally, you will learn how the PIX firewall can act as a concentrator for terminating Cisco software VPN clients.

IPsec Concepts

One of the technologies used to create VPNs is IPsec, which was developed by the Internet Engineering Task Force (IETF) as part of IPv6 and can be implemented in IPv4. IPsec is a framework of open standards that operates at Layer 3 of the OSI model, which means that it can protect communications from the network layer (IP) and up. The IPsec standards documents (of which there are many) can be found at www.ietf.org/html.charters/ipsec-charter.html. If you are interested in a detailed organization of the IPsec framework, it could be useful to start from RFC 2411 ("IP Security Document Road Map"), which describes the organization of the standards documents.

The overall technology concept is described in RFC 2401 ("Security Architecture for the Internet Protocol"). IPsec provides two security protocols used for transferring data, Encapsulating Security Payload (ESP) and Authentication Header (AH), which are described in RFC 2406 and RFC 2402, respectively. These documents describe the protocols, their corresponding packet structures, and implementation algorithms.

The encryption algorithm documents describe the encryption algorithms that are used by ESP implementations. The only required encryption algorithm in an IPsec implementation is Data Encryption Standard (DES), which is defined in RFC 1829. At this time, DES is considered inadequate protection and is being phased out in favor of stronger encryption such as Triple DES (3DES), Advanced Encryption Standard (AES), and Blowfish. To provide authentication features, IPsec uses the two algorithms HMAC-SHA-1 and HMAC-MD5, which are described in RFC 2404 and RFC 2403, respectively.

IPsec

IPsec's main design goals are to provide the follow functionality:

- **Data confidentiality** Data is encrypted before being transmitted, so nobody except the communicating parties can read it.
- **Data integrity** Each peer can determine if a received packet was changed during transit.
- **Data origin authentication** As an additional feature of data integrity service, the receiver can also check the identity of a packet's sender.
- **Antireplay** The receiver can detect and reject replayed packets, protecting it from spoofing and man-in-the-middle attacks.

IPsec Core Layer 3 Protocols: ESP and AH

As mentioned previously, IPsec was designed to provide confidentiality and integrity of transmitted information, authentication of participating parties, and protection against traffic replay. Two main network protocols, ESP and AH, are used to achieve this goal. All other parts of the IPsec standard are merely means of effectively implementing these protocols and configuring the required technical parameters. Applying AH or ESP to an IP packet means that the data part of the packet contents may be modified, although not always, and an extra header is inserted between the IP header and the packet contents. See Figures 7.1 and 7.2 for illustrations of how these transformations are performed.

Figure 7.1 AH Encapsulation

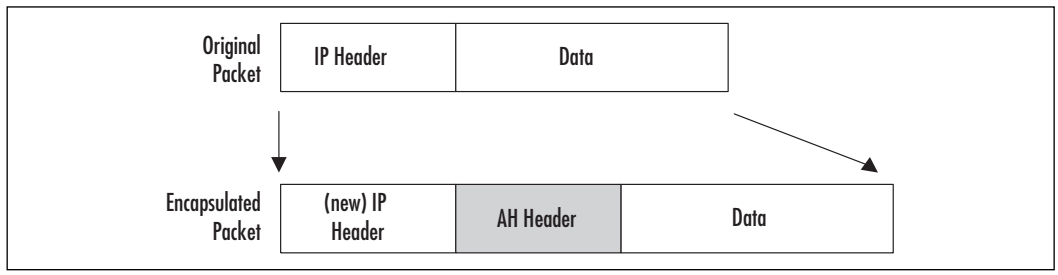
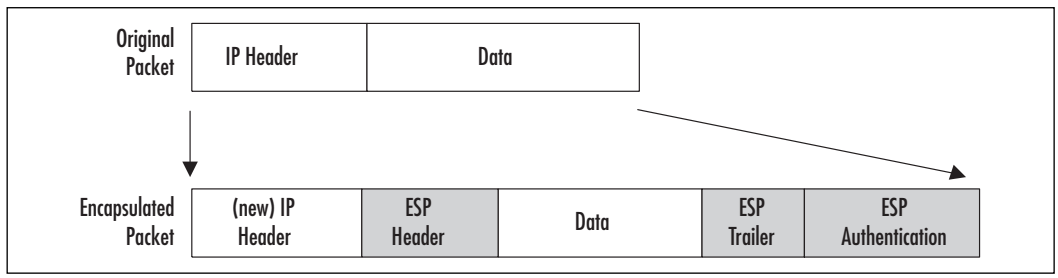


Figure 7.2 ESP Encapsulation



Authentication Header

The AH, which is defined as IP 51, is used to ensure the following:

- **Data integrity** This is achieved by calculating a hash of the entire IP packet, including the original IP header (not including variable fields such as the TTL), the data part of the packet, and the authentication header (excluding the field that will contain the calculated hash value). This hash is called an integrity check value (ICV), and it can be either Message Authentication Code (MAC) or a digital signature. MACs are more common than digital signatures. Hashing algorithms include MD5 and SHA-1, and both are known as *keyed hashes*, meaning that they use an extra value to calculate the hash, which is known only to the participating parties. When the packet is received, its content, excluding some fields, is hashed by the receiver and the result is compared with the ICV. If they are the same, the packet is declared authentic.
- **Data origin authentication** As part of the integrity feature, AH also provides source IP authentication. Since the source IP is included in the data, its integrity is guaranteed.

- **Replay protection** AH also includes an IPsec sequence number, which provides protection against replay attacks because this number is also included in authenticated data and can be checked by the receiving party.

AH provides no confidentiality because no encryption is used.

NOTE

Pure AH is *always* broken by NAT. For example, when an authenticated packet goes through an address-translation device, the IP address in its header changes and the MAC calculated by the receiver on a new packet will be incorrect, so the packet will be rejected. It is not possible for a translating gateway to recalculate the new MAC and insert it into the packet, because only the endpoints of a transmission know the hashing keys. This is a common problem with IPsec—trying to use AH when there is a NAT device somewhere in the path. It will simply *not* work. Use ESP with its own authentication (it is possible to turn on encryption if you want), or do not use NAT if you want to stay with AH.

Encapsulating Security Payload

ESP, which is defined as IP 50, provides the following features:

- Padding of a packet's contents in order to prevent traffic analysis, encryption of the result using ciphers such as DES, 3DES, AES, or Blowfish.
- Optional authentication using the same algorithms as the AH protocol. IP header information is not included in the authenticated data, which allows ESP-protected packets to pass through NAT devices without problems. When a packet is created, authentication data is calculated after encryption. This allows the receiver to check the packet's authenticity before starting the computationally intensive task of decryption.
- Optional antireplay features.

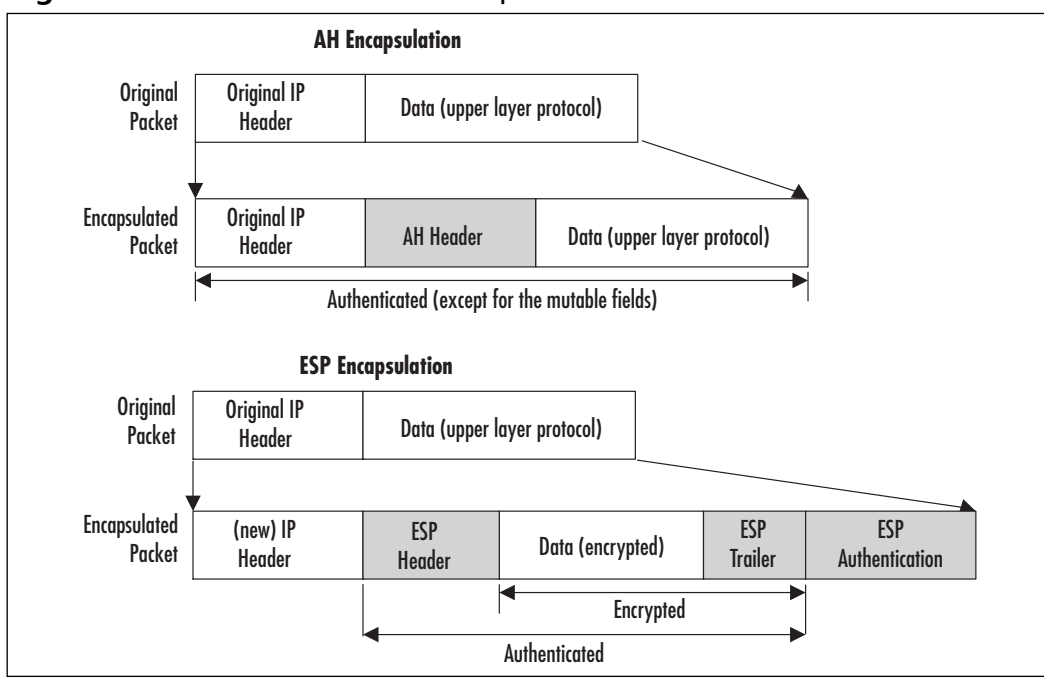
The original ESP definition did not include the last two features. It was assumed that the sender and receiver would use either one or both protocols at the same time if they needed confidentiality *and* authentication. Now, since ESP can also perform most of AH's features, AH is rarely used. Because ESP works on encapsulation principles, it has a different format: All data is encrypted and then placed between a header and a trailer. This differentiates it from AH, where only a header is created.

IPsec Communication Modes: Tunnel and Transport

Both AH and ESP can be applied in two modes: transport and tunnel. In transport mode, only the data portion of an IP packet is affected; the original IP header is not changed. Tunnel mode encapsulates the entire original packet as the data portion of a new packet and creates a new external IP header. (AH and/or ESP headers are created in both modes.) Transport mode is used when both the receiver and the sender are endpoints of the communication—for example, two hosts communicating directly to each other. Tunnel mode is more convenient for site-to site VPNs because it allows tunneling of traffic through the channel established between two gateways.

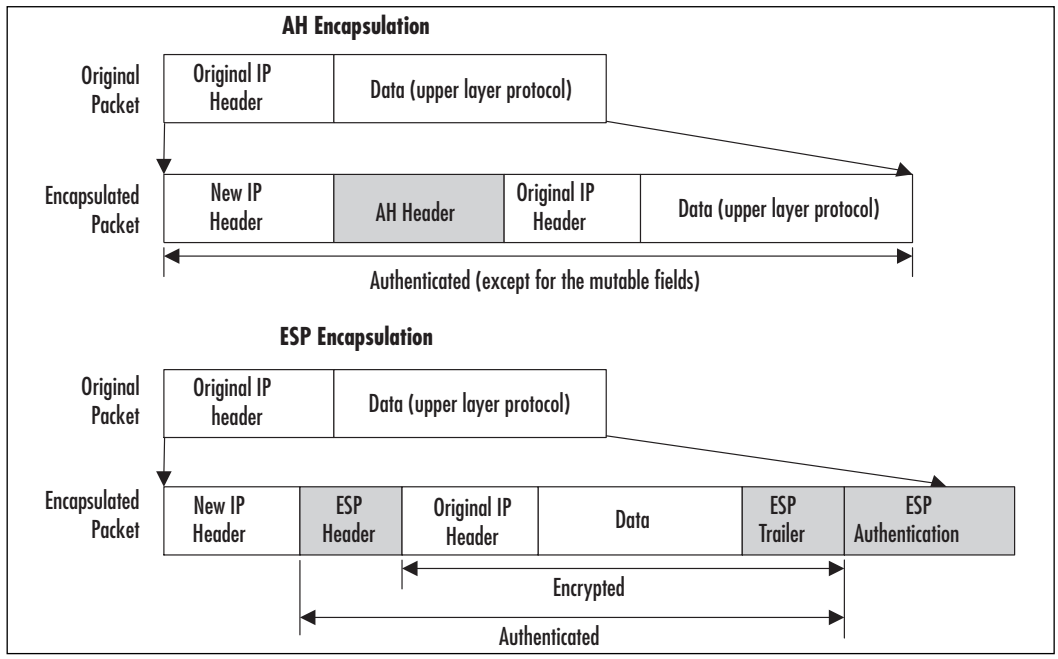
In transport mode, the IP packet contains an AH or ESP header right after the original IP header and before upper-layer data such as a TCP header and application data. If ESP is applied to the packet, only this upper-layer data is encrypted. If optional ESP authentication is used, only upper-layer data, not the IP header, is authenticated. If AH is applied to the packet, both the original IP header and upper-layer data are authenticated. Figure 7.3 shows what happens to the packet when IPsec is applied in transport mode.

Figure 7.3 Packet Structure in Transport Mode



Tunnel mode, the most common mode of operation, allows the establishment of an encrypted and authenticated IP tunnel between two sites. The original packet is encrypted and/or authenticated and encapsulated by a sending gateway into the data part of a new IP packet, and then the new IP header is added to it with the destination address of the receiving gateway. The ESP and/or AH header is inserted between this new header and the data portion. The receiving gateway performs decryption and authentication of the packet, extracts the original IP packet (including the original source/destination IPs), and forwards it to the destination network. Figure 7.4 demonstrates the encapsulation performed in tunnel mode.

Figure 7.4 Packet Structure in Tunnel Mode



Again, if AH is used, both the original IP header and the new IP header are protected (authenticated), but if ESP is used, even with the authentication option, only the original IP address, not the sending gateway's IP address, is protected. This setup is actually not that bad, because it is very difficult to spoof a correct IPsec packet without knowing many technical parameters. The exclusion of the new IP header from authenticated data also allows tunnels to pass through devices that perform NAT. When the new header is created, most of the options from the original IP header are mapped onto the new one—for example, the Type of Service (ToS) field.

Internet Key Exchange

The previous section described how network layer IPsec protocols function and which data they use. These protocols use cryptographic algorithms for encryption and authentication; thus some of the most important pieces of data are encryption/authentication keys. It is possible to configure these keys manually, but there are big disadvantages to this approach. First, it is very difficult to scale; second, it is not possible to renegotiate SAs because they are fixed until manually changed. Thus, there is a strong need of tools for managing keys and SAs. Key management includes generation, distribution, storage, and deletion of the keys. The most challenging phases are the initial authentication of the systems to each other and then the protection of the key exchange. After keys are exchanged, the channel is protected with these keys and used for setting up other parameters, including SAs.

The protocol the IETF adopted for performing these functions is called Internet Security Association and Key Management Protocol (ISAKMP), which is defined in RFC 2408 and describes the authenticated key exchange methods without diving into particularities. ISAKMP has an IANA-assigned UDP port number of 500. This is a generic protocol and is not tied to IPsec or any other key-using protocol. It can be implemented directly over IP or any transport layer protocol. When it is combined with parts of other key management protocols called Oakley (RFC 2412) and Secure Key Exchange Mechanism (SKEME), which has no RFC, we end up with a protocol called the Internet Key Exchange (IKE), which is defined in RFC 2409. Although not strictly correct, the abbreviations IKE and ISAKMP are often used interchangeably, even in Cisco configuration commands. In fact, on the PIX firewall, all IKE configuration is performed using the *isakmp* command.

In IKE, there are two phases of information exchange; each of them can operate in one or two modes. IKE Phase 1 starts when two peers need to establish a secure channel—that is, they do not have IPsec SAs needed for communication over IPsec. This phase includes authentication of systems by each other, agreement on encryption and authentication algorithms used from then on to protect IKE traffic, performing a Diffie-Hellman (DH) key exchange, and finally, establishing an IKE Security Association (IKE SA). IKE SAs are bidirectional; each IKE connection between peers has only one IKE SA associated with it. The second phase is centered on negotiating one or more IPsec SAs, which will be used for the IPsec tunnel between these peers. It uses key material from IKE Phase 1 to derive keys for IPsec. One peer tells the other which traffic it wants to protect and which encryption/authentication algorithms are supported. The

second peer then agrees on a single protection set for this traffic and establishes keys needed for this protection set.

Although implementing different phases adds some overhead in processing, there are advantages to this approach:

- Trust between peers is established in the first phase and used in the second phase.
- Key material established in the first phase can be used in the second phase.
- Renegotiations of the first phase can be assisted by the second-phase data.

Let's consider these two phases in more detail. Phase 1 has two modes: main mode and aggressive mode. Main mode uses three exchanges between peers; each exchange consists of two messages, a request and a reply:

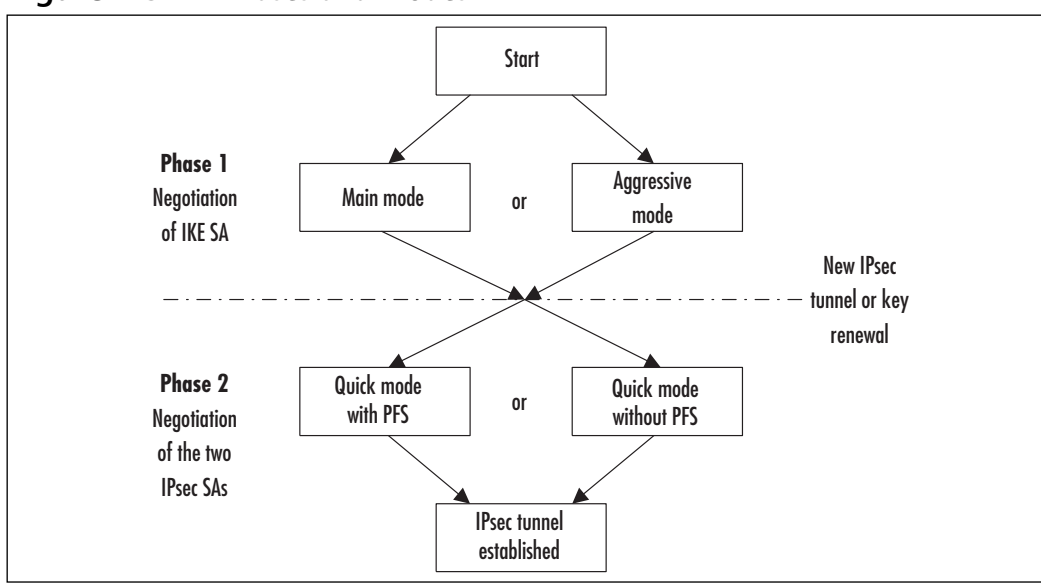
- The first exchange in main mode is used to negotiate the parameters for protection of the IKE connection. The initiating side sends a proposition to its counterpart, which includes a set of possible parameters supported by the initiator. Each set includes one encryption algorithm (DES, 3DES, etc.) and one of four authentication algorithms: pre-shared secret, RSA public key encryption with Diffie-Hellman exchange group 1 and 2, or public key RSA signature (this includes use of certificates). The other peer then replies by accepting a single pair from the offered set, which it also supports. If there is no match between the sets supported by the peers, the IKE tunnel cannot be established.
- The second exchange in main mode is used to perform DH key establishment between peers. It also exchanges two values called *nonces*, which are hashes that only the other part can interpret. This is done in order to confirm that the message is sent by the same hosts as the previous exchange.
- The third and last exchange performs actual authentication of the peers using the agreed-on methods: public keys signatures, public key encryption, or a pre-shared secret. This exchange is also protected by an encryption method that was selected in the first exchange.

RFC 2408 provides more details on the packet format and algorithms used. At the end of the first phase, each host has an IKE SA, which specifies all parameters for this IKE tunnel: the authentication method, the encryption and hashing algorithm, the Diffie-Hellman group used, the lifetime for this IKE SA, and the key values.

Aggressive mode exchanges only three packets instead of six, so it is faster but not as secure. The number of packets is decreased because the first two packets in this exchange include almost everything in one message; each host sends a proposed protection set, Diffie-Hellman values and authentication values. The third packet is sent only for confirmation and after the IKE SA is already established. The weakness in aggressive mode is that everything travels on the wire in clear text and can be eavesdropped or spoofed. However, the only thing the attacker can achieve is to DoS one of the peers, because it is not possible to discover the keys that are established by the Diffie-Hellman protocol.

The most important mode of Phase 2 is quick mode. It can be repeated several times using the same IKE SA established in Phase 1. Each exchange in this mode results in the establishment of two IPsec SAs by each peer. One of these SAs is used for inbound protection, and the other is used for outbound protection. During the exchange, peers agree on the IPsec SA parameters and send each other a new nonce, which is used for deriving Diffie-Hellman keys from the ones established in Phase 1. When the IPsec SA lifetime expires, a new SA is negotiated in the same manner. Figure 7.5 summarizes the flow of the IKE protocol.

Figure 7.5 IKE Phases and Modes



Another mode in Phase 2 is new group mode, which is not related to the setup of IPsec parameters and is used to change parameters of the Diffie-Hellman group used in IKE Phase 1.

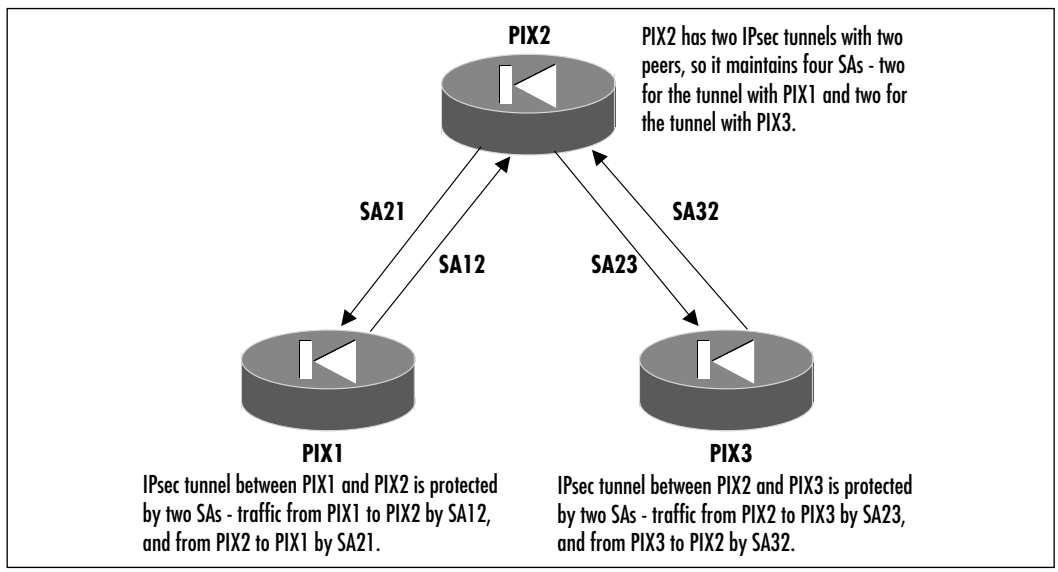
NOTE

It is possible to request that quick mode use Perfect Forward Secrecy (PFS). PFS dictates that new encryption keys are not derived from previous ones, so even if one key is discovered, only the traffic protected by this key and nothing else will be exposed. PFS is achieved by performing a new Diffie-Hellman key establishment in each quick mode.

Security Associations

All previous descriptions of protocols' functionality were based on a presumption that an IPsec connection is already established and all parameters such as authentication and encryption keys are known to both parties. Let's see how these parameters are managed in IPsec framework. The data flow in each direction is associated with an entity called a *security association (SA)* or, more specifically, an IPsec SA. This means that in a two-way communication, each party has at least two IPsec SAs: The sender has one for outgoing packets and another for incoming packets from the receiver, and the receiver has one SA for incoming packets from the sender and a second SA for outgoing packets to the sender. See Figure 7.6 for an illustration.

Figure 7.6 IPsec Security Associations and Their Use in Two-Ways Communication



Each SA can be uniquely identified by three parameters:

- The Security Parameter Index (SPI), which is always present in AH and ESP headers
- The destination IP address
- The IPsec protocol, AH or ESP (so if both protocols are used in communication, each has to have its own SA, resulting in a total of four SAs for two-way communication)

Each participating host or gateway maintains a separate database of active SAs for each direction (inbound and outbound) on each of its interfaces. This database is known as the Security Association Database (SAD). SAs from these databases decide which encryption and authentication parameters are applied to the sent or received packet. SAs may be fixed for the time of traffic flow (called *manual IPsec* in some documents), but when a key management protocol is used, they are renegotiated many times during the connection flow. For each SA, the SAD entry contains the following data:

1. The destination address
2. The SPI
3. The IPsec transform (protocol and algorithm used—for example, AH, HMAC-MD5)
4. The key used in the algorithm
5. The IPsec mode (tunnel or transport)
6. The SA lifetime (in kilobytes or in seconds); when this lifetime expires, the SA must be terminated, and a new SA established
7. The antireply sequence counters
8. Some extra parameters such as Path MTU

The selection of encryption parameters and corresponding SAs is governed by another database, the Security Policy Database (SPD). An SPD is maintained for each interface and is used to decide on the following:

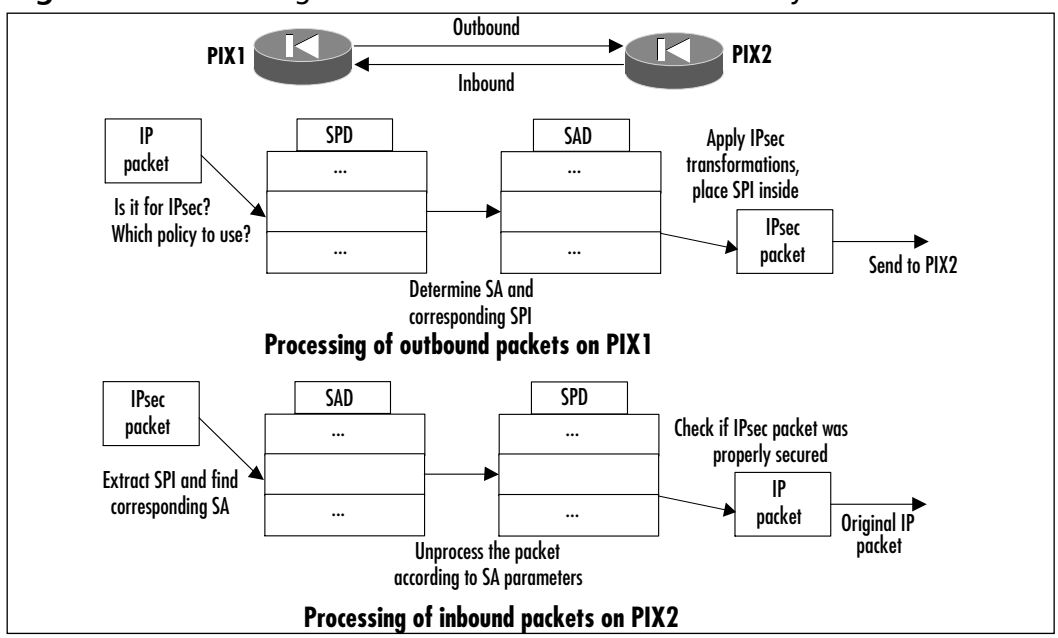
- Selection of outgoing traffic to be protected
- Checking if incoming traffic was properly protected
- The SAs to use for protecting this traffic
- What to do if the SA for this traffic does not exist

The SPD consists of a numbered list of policies. Each policy is associated with one or more selectors. A selector in Cisco's implementation is simply an access list. A *permit* statement means that IPsec should be applied to the matching traffic; a *deny* statement means that the packet should be forwarded and IPsec not applied. SPD policies are configured on the PIX firewall with the *crypto map* command. The resulting map and a crypto access list are applied to the interface, creating an SPD for this interface.

For outgoing traffic, when the IPsec network stack layer receives data to be sent, it consults the SPD to check if the traffic has to be protected. If it does, the SPD is used to recover an SA that corresponds to this traffic. If the SA exists, its characteristics are taken from the SAD and applied to the packet. If the SA does not exist yet, IKE is called upon to establish a new SA, and then the packet is protected with characteristics of this SA.

For incoming IPsec traffic, the SPI is recovered from AH or ESP header, then it is used to find a corresponding SA in SAD. If it does not exist, the packet is dropped. If an SA exists, the packet is checked/decrypted using the parameters provided by this SA. Finally, the SPD is checked in order to ensure that this packet was correctly protected—for example, that it should have been encrypted using 3DES and authenticated with MD5 and nothing else. Figure 7.7 shows both sequences of events.

Figure 7.7 Processing of Outbound and Inbound Traffic by IPsec



Designing & Planning…

Cryptographic Algorithms in IPsec and Their Relative Strengths

This section mentions many cryptography-related terms. Three categories of cryptography algorithms are used in all IPsec implementations:

- Encryption algorithms
- Message authentication algorithms
- Key establishment algorithms.

Encryption algorithms are used for enciphering clear-text messages, turning them into cipher text and deciphering them back to normal state using cryptographic keys. The simplest type of encryption algorithms use *symmetric encryption*. In this case, messages can be decrypted using the same key with which they were encrypted, and vice versa. This key must be kept a secret and well protected; otherwise, anybody can read and create encrypted messages. In addition, a general rule is that the longer the key, the more difficult it is to "crack" an encrypted message without knowing the key.

An example of this type of encryption is DES. It was adopted by the U.S. government as an official standard until it was recently replaced by Advanced Encryption Standard (AES), which provides much stronger encryption. DES is now considered obsolete and weak because the speed of computers has increased to the point that messages encrypted with standard 56-bit DES can easily be cracked.

A stronger variation of DES is Triple DES (3DES). It encrypts a message three times using DES, each time using a different 56-bit key. 3DES is still considered a strong cipher, although in a few years will be phased out in favor of AES. The PIX firewall supports DES and 3DES as encryption algorithms for IPsec. The corresponding keywords for ESP configuration are *esp-des* and *esp-3des*. When configuring IKE, the keywords are *des* and *3des*.

Another type of encryption is public-key cryptography. It uses complex exponential calculations and is rather slow compared with fast symmetric-key ciphers such as DES or 3DES. The basic advantage of public-key cryptography is that it uses two keys: one for encryption and a completely separate one for decryption. Only the decryption key (known as the *private key*) needs to be kept secret. The encryption key

Continued

(known as the *public key*) can be made public. For example, if anyone wants to send Alice an encrypted message, they can use her public key to encrypt the message, but only Alice knows the key that allows her to decrypt the message. One widespread algorithm based on public keys is the Rivest, Shamir, and Adelman (RSA) algorithm.

Message authentication algorithms are used to protect the integrity of a message. IPsec uses two types: keyed message hash algorithms and public signature algorithms. *Keyed message hashing* works in the following manner: A message is combined with a key and then reduced to a fixed-length digest. (Adding a key gives these algorithms the name *keyed*.) A *hashing algorithm* has a specific property, which makes it almost impossible to create a message with the same digest as a given one. When a receiver wants to ensure that the message was not altered in transit, it performs the same calculation on the message and compares the result with the received digest. If they are the same, the message is authentic; a spoofed one would have a different digest.

The two authentication algorithms IPsec uses are MD5, which produces 128-bit output, and SHA-1, which produces 160-bit output and is stronger than MD5. Although SHA-1 is cryptographically stronger than MD5, it requires more processing to compute the hash than MD5. IPsec uses modified versions of these authentication algorithms, called HMAC-MD5 and HMAC-SHA-1, which perform hashing twice, each time combining in a different way the message to be digested with the key. The PIX firewall supports both HMAC-MD5 and HMAC-SHA-1.

Finally, key establishment protocols provide means for secure exchange of symmetric keys by both sides via an insecure medium (such as the Internet). In IPsec, this task is accomplished using the Diffie-Hellman (DH) algorithm. DH is based on some exponential computations, and during the process both sides exchange a couple of numbers, allowing both peers to derive the same key, but nobody who sees these numbers can do the same. DH in IPsec can work with keys of two different lengths: 768-bit (DH Group 1) and 1024-bit (DH Group 2). Although Group 2 keys are stronger, they require much more processing power. The PIX firewall supports both types of DH keys, with Group 1 being the default choice.

Certificate Authority Support

IKE authentication on the PIX firewall can be performed in two different ways:

- Using pre-shared keys, where the parties simply send each other a value—their own names, for example, which are encrypted using the shared key and a hash of some parameters
- Using RSA signature authentication (digital certificates)

In the second method, each party, in order to identify itself, will send to the other the following set of values: its name, its public certificate issued by a certificate authority (CA), and its RSA signature. A public key certificate contains a copy of the party's public key. The receiving party queries the same CA (of course, this CA should be trusted by the receiving party) and checks to see if the certificate really belongs to the sender. If it does, the RSA signature is verified using the public key from the certificate, and the system's identity is verified. The biggest advantage of using certificate authorities for authentication in IKE is that this scheme is easily scalable, especially in partial- or full-mesh environments. When a new peer is added to the IPsec network, the administrator only needs to enroll it with the CA and obtain a certificate from the CA. After that, each participant that recognizes this CA will be able to verify the identity of the new peer by its certificate.

In order to receive a certificate, a system must establish a trusted channel with the CA, generate a public/private key pair, and request a certificate. The CA then verifies the system's credentials somehow (usually using offline methods) and issues a certificate. A certificate can include a good deal of information: the bearer's IP address, its name, the serial number of the certificate, the expiry date of the certificate, and a copy of the bearer's public key. The standard for the certificate format is X.509, and Cisco supports version 3 of this standard. The PIX firewall requires that the CA support the Simple Certificate Enrollment Protocol (SCEP). Currently, the following CAs are supported:

- VeriSign Private Certificate Services (PCS) and On-Site service (www.verisign.com)
- Entrust VPN Connector version 4.1 or higher (www.entrust.com)
- Baltimore Technologies UniCERT Certificate Management System, version 3.1.2 or higher
- Microsoft CA, a part of Microsoft Windows 2000 Advanced server

Configuring Site-to-Site IPsec Using IKE

At a high level, the IPsec configuration process on the Cisco PIX firewall consists of three major steps:

1. **Planning** Deciding on the details of IPsec policies used, such as the SA establishment method (if IKE is used or SAs are configured manually), IKE parameters, including the peer authentication methods (using pre-shared keys or digital certificates), the protocols that will be used (ESP amdpr AH) and in which modes, and the encryption algorithms. This step also includes ensuring that the peers are able to communicate without IPsec and that all IPsec packets are allowed to bypass ordinary access lists and conduits.
2. **Configuring IKE (if used)** This step includes enabling IKE on the firewall, configuring policy parameters for Phases 1 and 2, and defining the authentication method (pre-shared keys or CA).
3. **Configuring IPsec parameters** This step includes defining interesting traffic, configuring transform sets, creating a crypto map, and applying this map to an interface.

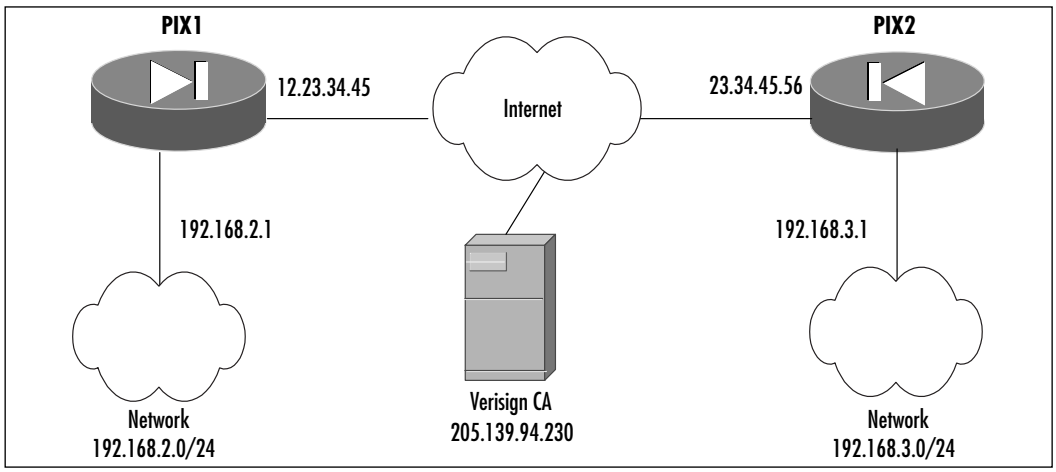
Using an example, let's go through the configuration of a site-to-site IPsec VPN using IKE. We also discuss the differences between using pre-shared keys and digital certificates.

Planning

Applying the three-step plan outlined, we first need to do some planning. Let's decide on the configuration parameters. Figure 7.8 shows the networks and IP addresses that are used in the example.

First, we need to decide on IKE Phase 1 parameters. Protection parameters include the peer authentication method (pre-shared keys or digital certificates), the encryption algorithm (DES or 3DES), the data authentication algorithm (MD5 or SHA-1), the DH group identifier (Group 1 or Group 2), and the IKE SA lifetime. All these parameters together constitute an IKE policy. It is possible to configure a different set of policies for each remote peer, but at least one policy must be shared by both firewalls in order for the IKE tunnel to be established. In this example, we use 3DES encryption, MD5 authentication, DH Group 2, and an IKE SA lifetime of 2400 seconds. We provide examples for using both pre-shared keys as well as digital certificates as the authentication method. If using pre-shared keys, we must determine the keys to use. We use the key string *mykey1*.

Figure 7.8 Network Setup for a Site-to-Site VPN



The next task is the selection of IPsec parameters. Besides confirming peer IP addresses and names, we need to decide if the IPsec SAs will be created with the help of IKE, and we must select transform sets for each peer. Again, it is possible to configure many different transform sets for each IPsec tunnel, but at least one must be the same on both firewalls in order for an IPsec SA and an IPsec tunnel to be successfully established. In this example, we configure a transform set with tunnel mode, ESP protection with DES, and ESP authentication with SHA-1.

Now we are ready for configuration. Let's go through it step by step. Please note that the steps *defining an ISAKMP pre-shared key* and *configuring certificate authority support* are exclusive, and only one of them needs to be performed.

Allowing IPsec Traffic

The first step in configuration is to confirm that the two firewalls can reach each other before IPsec is turned on. Ping each firewall from the other, and ensure that there is network connectivity. Of course, if ICMP is disabled, the pings will not work.

The next step is to permit incoming IPsec traffic to reach the firewall. There are two different ways of doing this. The first is to use the *sysopt connection permit-ipsec* command, which implicitly allows all IPsec-related traffic to reach the firewall. This is equivalent to adding the following lines to the access list on the outside PIX interface:

```
PIX1(config)# access-list outside_access_in permit 50 any host 12.23.34.45
PIX1(config)# access-list outside_access_in permit 51 any host 12.23.34.45
PIX1(config)# access-list outside_access_in permit udp any host 12.23.34
    .45 eq 500
```

The first two lines allow any traffic with IP 50 (ESP) and 51 (AH) to reach the outside interface, and the third allows IKE traffic, which is directed to UDP port 500. Instead of using the *sysopt* command, we can create more granular access control for each firewall using access lists or conduits, which are the second way to permit IPsec traffic. For example, the following access list allows IPsec traffic only from PIX2 reach PIX1:

```
PIX1(config)# access-list outside_access_in permit 50 host 23.34.45.56
    host 12.23.34.45
PIX1(config)# access-list outside_access_in permit 51 host 23.34.45.56
    host 12.23.34.45
PIX1(config)# access-list outside_access_in permit udp host 23.34.45.56
    host 12.23.34.45 eq 500
```

Configuring the *sysopt connection permit-ipsec* command is the preferred method of allowing IPsec traffic, because it is simpler and does not really open any holes in the firewall. Since IPsec packets are encrypted and authenticated, any packet that does not come from a correct peer will be discarded. However, if you do not use this *sysopt* command, do not forget to create access lists on the outside interface (or another interface at which the tunnel terminates) to permit the traffic you need. With the *sysopt* command, all decapsulated IPsec traffic is allowed to pass through without additional conduits.

NOTE

It is useful to check that all network devices between the two firewalls are configured to pass traffic with IP 50 and 51 and UDP traffic with a destination port 500. Some providers have an acceptable use policy (AUP) that does not allow VPN, so they filter IPsec. Others only allow IPsec traffic to pass through as a value-added service for those customers that want to use an IPsec VPN and are willing to pay for it.

Enabling IKE

Configuration of IKE policies starts with enabling IKE processing on the outside interface of the firewall (or any other interface that is connected to the remote peer). This must be done on each peer using the following command:

```
isakmp enable <interface_name>
```

In our example, this command needs to be on the outside interface of each firewall:

```
PIX1(config)# isakmp enable outside
PIX2(config)# isakmp enable outside
```

IKE is enabled on all interfaces by default. It can be turned off on a specific interface (to prevent DoS attacks on the interface) using the *no* form of the command:

```
no isakmp enable <interface_name>
```

By default, the PIX firewall uses its IP addresses to identify itself to its peers. The PIX can identify itself (and its peers) by either an IP address or a hostname. When peers are to be authenticated by RSA signatures, it is recommended that the hostname be used. (The remote peer must either be define on the firewall using the *name* command, or it must be resolvable through DNS.) On the other hand, if you requested digital certificates that include IP addresses, you should stick with the default of using the IP address for the identity method. To change the identity method, use the following command, but be sure to use the same method on both firewalls:

```
isakmp identity {address | hostname}
```

If the identity method does not match, the peers will not be able to negotiate an IKE SA and thus no IPsec SA will be established.

Creating an ISAKMP Protection Suite

The next step is to configure IKE policy parameters. The PIX can have many IKE policies (also known as *ISAKMP protection suites*), which are distinguished by their priority (an integer from 1 to 65,534). The smaller this number, the higher the policy's priority. The IKE policy parameters between peers must match exactly. A policy with the smallest number is attempted first, and then if it is not accepted by the remote peer, the next is attempted. This process continues until

one of the policies is accepted by the other peer or the policy list is exhausted and IKE establishment fails. To create a policy, use the following commands:

```
isakmp policy <priority> authentication {pre-share | rsa-sig}
isakmp policy <priority> encryption {des | 3des}
isakmp policy <priority> hash {md5 | sha}
isakmp policy <priority> group {1 | 2}
isakmp policy <priority> lifetime <lifetime>
```

These commands specify (in order) the encryption algorithm to be used, the data authentication algorithm, the peer authentication method, the Diffie-Hellman group identifier, and the IKE SA lifetime in seconds. The lifetime can be any number of seconds between 2 and 3600.

According to our plan, we will configure the following on both firewalls using a priority of 10:

```
isakmp policy 10 encryption 3des
isakmp policy 10 hash md5
isakmp policy 10 group 2
isakmp policy 10 lifetime 2400
```

If any of these parameters is not specified, the default value is used. The default values for each of these parameters are *des* for encryption, *md5* for data authentication, *1* for DH group, and *3600* for IKE SA lifetime. Of course, we must also specify the peer authentication method. If you are using pre-shared keys, use the following command:

```
isakmp policy 10 authentication pre-share
```

If you are using digital certificates, use the following command (although it is the default and does not really need to be specified).

```
isakmp policy 10 authentication rsa-sig
```

To verify the configuration of IKE policies, use the *show isakmp policy* command. If you're using pre-shared keys, the output should be as follows:

```
PIX1# show isakmp policy
Protection suite of priority 10
      encryption algorithm:   Three key triple DES
      hash algorithm:          Message Digest 5
      authentication method:  Pre-Shared Key
      Diffie-Hellman group:   #2 (1024 bit)
```

```
      lifetime:                2400 seconds, no volume limit
Default protection suite
      encryption algorithm:   DES - Data Encryption Standard (56 bit keys).
      hash algorithm:          Secure Hash Standard
      authentication method:  Rivest-Shamir-Adleman Signature
      Diffie-Hellman group:   #1 (768 bit)
      lifetime:                86400 seconds, no volume limit
```

As you can see here, there is also a default IKE policy. Although it cannot be seen in the output of the command, this default policy has a priority of 65,535. If the configured ISAKMP policies do not match a proposal by the remote peer, the firewall tries this default policy. If the default policy also does not match, ISAKMP negotiation fails.

Defining an ISAKMP Pre-Shared Key

The most common site-to-site VPN setup between two PIX firewalls is the configuration of an IPsec tunnel with IKE using a pre-shared key. If you use the firewall to establish a number of VPNs with different peers, it is highly recommended that the pre-shared key be unique for each pair of gateways. The key to be used for establishing an IKE tunnel with the particular peer is selected based on the peer's IP address. The key itself is an alphanumeric string of up to 128 symbols and must be configured the same on both gateways using the following command:

```
isakmp key <keystring> address <peer-address> netmask [netmask]
```

We need to configure the key on both firewalls:

```
PIX1(config)# isakmp key mykey1 address 23.34.45.56 netmask 255.255.255
    .255
PIX2(config)# isakmp key mykey1 address 12.23.34.45 netmask 255.255.255
    .255
```

In order to use the same key for connecting to any peer, use 0.0.0.0 both as a peer address and as a netmask.

Configuring Certificate Authority Support

Use of CAs is very helpful when you need to configure a large and scalable network of interconnected peers, where peers can be added or removed at any time. If you configured a network with IKE using pre-shared keys, you would need to

change the configuration of several firewalls each time a new one is added or removed. CAs provide an easy method for configuring complicated networks. The main advantage is that each peer is configured separately and independently from others. When public key certificates are used for authenticating parties in IKE, each peer has a certificate of its own and presents it to its counterpart during the IKE authentication phase. The other side verifies the authenticity and validity of this certificate by consulting a CA and, if everything is all right, IKE authentication is successful. The CA can either be a machine available on your network or you can use a trusted external authority. In our example, we use an external VeriSign server that has an IP address of 205.139.94.230.

Enrollment is a complex process and includes the following steps:

1. The PIX generates its own RSA public/private key pair.
2. The PIX requests the CA's public key and certificate. This must either be done over a secure channel or be checked by some offline means—for example, by comparing certificate fingerprints.
3. The PIX submits a request for a new certificate. This request includes the public key generated at Step 1 and is encrypted with the CA's public key obtained in Step 2.
4. The CA's administrator verifies the requester's identity and sends out a new certificate. This certificate is signed by the CA, so its authenticity can be verified by anybody who has a copy of the CA's certificate.

NOTE

Before configuring CA support on the PIX, make sure that its internal clock and time zone have been set correctly.

You need to decide if you will be using certificate revocation lists (CRLs). These lists are maintained by some CAs as means of checking for revoked certificates. If you turn on CRL support, before each certificate is accepted it will be checked against this list. This requires that a connection between the firewall and the CA must be available at the time of authentication, which is not always possible. If you do not use CRLs, you only need connectivity with CA during enrollment, and all authentication of certificates afterward is done using the CA's public certificate, which the firewall obtained from CA during enrollment.

Configuring the Hostname and Domain Name

The enrollment starts with defining the firewall's hostname and domain name, which will be used in its certificate later. As discussed in Chapter 2, the commands to configure these are:

```
hostname <hostname>
domain-name <domain-name>
```

In our example, we need to enter the following commands:

```
PIX1(config)# hostname PIX1
PIX1(config)# domain-name securecorp.com
PIX2(config)# hostname PIX2
PIX2(config)# domain-name securecorp.com
```

Generating an RSA Key Pair

The next step is the creation of an RSA key pair. This is achieved by using the following command:

```
ca generate rsa key <key_modulus_size>
```

This command forces the PIX to generate a public/private RSA key pair and store it in flash memory. The strength of the generated keys is specified using the *key_modulus_size* parameter. The default value is 768 bits, which rather secure, but you can use 1024 or 2048 bits if you like. Be sure that you have correctly configured host and domain names for the PIX before you generate the keys. For example:

```
PIX1(config)# ca generate rsa key 1024
Key name:PIX1.securecorp.com
Usage:General Purpose Key
Key Data:
30819f30 0d06092a 864886f7 0d010101 05000381 8d003081 89028181 00c8ed4c
9f5e0b52 aea931df 04db2872 5c4c0afd 9bd0920b 5e30de82 63d834ac f2e1db1f
1047481a 17be5a01 851835f6 18af8e22 45304d53 12584b9c 2f48fad5 31e1be5a
bb2ddc46 2841b63b f92cb3f9 8de7cb01 d7ea4057 7bb44b4c a64a9cf0 efaacd42
e291e4ea 67efbf6c 90348b75 320d7fd3 c573037a ddb2dde8 00df782c 39020301
     0001
```

Generated keys are stored in flash memory. The public key can be viewed by issuing the command:

```
show ca mypubkey rsa key
```

For example:

```
PIX1(config)# show ca mypubkey rsa
% Key pair was generated at: 09:45:23 Sep 11 2002
Key name:PIX1.securecorp.com
Usage:General Purpose Key
Key Data:
30819f30 0d06092a 864886f7 0d010101 05000381 8d003081 89028181 00c8ed4c
9f5e0b52 aea931df 04db2872 5c4c0afd 9bd0920b 5e30de82 63d834ac f2e1db1f
1047481a 17be5a01 851835f6 18af8e22 45304d53 12584b9c 2f48fad5 31e1be5a
bb2ddc46 2841b63b f92cb3f9 8de7cb01 d7ea4057 7bb44b4c a64a9cf0 efaacd42
e291e4ea 67efbf6c 90348b75 320d7fd3 c573037a ddb2dde8 00df782c 39020301
   0001
```

The private key cannot be viewed.

Specifying a CA to Be Used

After the key pair is generated on the PIX firewall, we need to specify the CA to use for certificate verification. The command for doing so is:

```
ca identity <ca_nickname> <ca_ip_address>[:<script_location>] [<ldap_
    address>]
```

The *ca_nickname* parameter specifies an internal nickname that the PIX will use for this CA, and *ca_ip_address* specifies the IP address of the CA server. The *script_location* parameter can be specified when the CA uses a nonstandard URL for the enrollment script, which by default should reside at /cgi-bin/pkiclient.exe. For example, when using a Microsoft CA, specify */CERTSRV/mscep/mscep.dll*. If the CA supports LDAP requests, you can specify the IP address of CA's LDAP server in the command as well.

The PIX supports only one CA at a time. In order to remove a CA, simply use the following command:

```
no ca identity <ca_nickname>
```

For our example, we use the following configuration:

```
PIX1(config)# ca identity verisign 205.139.94.230
```

```
PIX2(config)# ca identity verisign 205.139.94.230
```

The CA identity settings can be verified using the *show ca identity* command.

Configuring CA Parameters

The next step, to configure CA parameters, is accomplished using the following command:

```
ca configure <ca_nickname> {ca|ra} <retry_period> <retry_count>
    [crloptional]
```

This command specifies whether *ca_nickname* is a CA or a registration authority (RA). Some systems use an RA, which the firewall uses instead of a CA. An RA is somewhat like a proxy for the CA but is rarely used in small-to-medium-sized networks. The command also specifies the number of retries that the PIX should perform when trying to contact this authority and the timeout between requests (in minutes). The *crloptional* parameter tells the PIX to skip checking certificates against the CRL if the CRL is unavailable. If *crloptional* is not specified but the CRL is unavailable, the peer's certificate will be rejected.

NOTE

Always use the *crloptional* parameter with both public and in-house versions of VeriSign CAs, because they do not provide a CRL at all.

We will use the following:

```
PIX1(config)# ca configure verisign ca 1 20 crloptional
PIX2(config)# ca configure verisign ca 1 20 crloptional
```

This means that the authority previously identified as *verisign* is a CA, it does not support CRLs, and the PIX should retry 20 times with the delay of 1 minute before giving up on the connection to this CA. To view the CA configuration settings, use the *show ca configure* command.

Authenticating the CA

The next step is obtaining the CA's public key and verifying its authenticity. This key is contained in the CA's own digital certificate, which is self-signed by the CA. Therefore, after obtaining this certificate, the PIX has to verify that it is using

some offline method. This can be achieved by obtaining a special characteristic of the certificate, a "fingerprint," from the CA's administrator (or by other means). A *fingerprint* is a hash of the certificate's content, and if the calculated hash and received hash match, the certificate is original. The command used on PIX for requesting the CA's certificate is:

```
ca authenticate <ca_nickname> [<fingerprint>]
```

If this command is used with only one parameter—the CA's nickname—the PIX simply requests the certificate from the CA and displays the results of this action:

```
PIX1(config)# ca authenticate verisign
Certificate has the following attributes:
Fingerprint: 1234 1234 5678 CDEF ABCD
```

The PIX also calculates a fingerprint of the received certificate (10 bytes in hexadecimal encoding) and displays it. It is possible then to compare it with the known fingerprint to verify authenticity of the certificate. The verification can be done automatically if the known fingerprint is entered as part of the command:

```
PIX1(config)# ca authenticate verisign 0123456789abcd012345
Certificate has the following attributes:
Fingerprint: 0123 4567 89AB CDEF 5432
%Error in verifying the received fingerprint. Type help or '?' for a list
    of available commands.
```

In this case, the calculated fingerprint (0123 4567 89AB CDEF 5432) and the expected one (0123 4567 89ab cd01 2345) did not match. So in this case, a certificate is discarded. The *ca authenticate* command is not stored in the PIX configuration; there is no need to perform it more than once for each new CA. If the authority you are using is an RA instead of a CA, it will return three certificates:

- The RA signing key
- The RA encryption key
- The CA general-purpose public key

The received certificate is stored in the memory area designated for storing the firewall's RSA keys (the whole record is called the *RSA public key chain*) and can be viewed with the following command:

```
show ca certificate
```

It produces output similar to this:

```
RA Signature Certificate
Status: Available
Certificate Serial Number: 38231245
Key Usage: Signature

CA Certificate
Status: Available
Certificate Serial Number: 38231256
Key Usage: Not Set

RA KeyEncipher Certificate
Status: Available
Certificate Serial Number: 38231267
Key Usage: Encryption
```

CA certificates must be stored in flash memory using the *ca save all* command or they will be lost after a reboot. The *write memory* command does not save certificates.

Enrolling with the CA

During the enrolling process, a firewall sends a request to the CA to issue a new certificate for this firewall. The CA will reply by signing the public key certificate, which it receives from the firewall as a part of the request and returning the results to the PIX. After the CA signs it, it becomes a valid certificate and its authenticity can be validated by usual public key signature tools by anyone who knows the CA's public key. Technically, the CA does not have to reply (issue a certificate) immediately and the certificate can be sent long after the request was sent (the enrollment process itself), but in practice the PIX expects these two events to happen during one transaction.

The enrollment is started by the following command:

```
ca enroll <ca_nickname> <challenge_password> [serial] [ip_address]
```

Here, the *ca_nickname* is a CA defined earlier using the *ca identity* and *ca authenticate* commands. The *challenge_password* parameter is a password that will be used to authenticate future requests for revoking a certificate. This means that if you later need to revoke the certificate obtained by this enrollment, you need to provide the CA with the same password that you specified during enrollment.

When the *ca authenticate* command is issued, the PIX requests one public key certificate for each of its RSA key pairs. If you generated only one pair of keys (using the *ca generate rsa key* command), a single certificate will be requested. If there are any more RSA pairs (for use with SSL for example—a special-use key pair), the PIX requests more certificates. If it already has been issued a certificate, the PIX will prompt you to delete existing certificates from its memory. Certificates can also be removed using the following command:

```
no ca identity <ca_nickname>
```

This command removes all certificates issued by the specified authority.

The *ca enroll* command, including the challenge password, is not stored in the PIX configuration; only its results can be stored in flash memory by the *ca save all* command.

The *serial* and *ip_address* options allow inclusion of some extra information in the public key certificate. When the *serial* option is specified, the firewall's serial number is included in the certificate request and, as a consequence, in the resulting certificate. This number is not used by IPsec or IKE, but it might be used later by the CA administrator for additional authentication. The second option is more important when IKE is used and has to do with device authentication. By default, when the *ip_address* option is not specified in the *ca enroll* command, a certificate is bound only to the host and domain names of the PIX device (a fully qualified domain name, or FQDN), which have to be specified prior to any CA-related configurations using the *hostname* and *domain* commands. If the *ip_address* option is specified, an IP address of the firewall is also included in the certificate. As a result, this certificate can be used only by the device with this IP address. If you move the firewall to a new address (even if its FQDN remains the same), you will need a new certificate.

NOTE

It is important that the IKE identity type is the same as the certificate type. This means that if you use default certificates, bound only to the FQDN, you need to set the IKE identity type to *hostname*:

```
isakmp identity hostname
```

The default setting for the IKE identity type is *address.* If you want to use IP addresses for authentication, specify *ipaddress* in the *ca enroll* command and set the identity type to *IP address*:

```
isakmp identity address
```

Back to our example: We will use the previously defined CA *verisign* and host-based authentication, so the enrollment in this case is very simple. (Remember that in this case we need to specify *isakmp identity hostname* in IKE configuration.) This command:

```
pix1(config)# ca enroll verisign midnightinmoscow
```

performs enrollment of PIX1 to CA *verisign* and sets the challenge password to *midnightinmoscow*. On the second firewall, we issue the following command, which performs the same operation on PIX2 but sets a different challenge password for the issued certificate:

```
PIX2(config)# ca enroll verisign lunchtimeinLA
```

It is possible to display obtained certificates on the firewall with the *show ca certificate* command. The example was shown in the previous section, "Authenticating the CA."

At this point, all CA-related information should be saved:

```
PIX1(config)# ca save all
PIX1(config)# write memory
```

Of all these *ca* commands, only *ca identity* and *ca configure* will be stored in the PIX configuration. The other commands just store their results, because there is no need to perform them when the firewall reboots.

Configuring Crypto Access Lists

The first stage in the process of IPsec configuration (or creating an SPD, in terms of the first section of this chapter) is specifying traffic selectors for IPsec. Selectors define which traffic will be protected by IPsec; to be precise, they will define the scope of SAs when they are created by IKE Phase 2. These selectors are defined using the *access-list* command. Crypto access lists are applied to the interface using a *crypto map* command instead of *access-group*. It is possible to apply many crypto access lists to one interface in order to specify different parameters for different types of traffic. Actions in access list entries have the following meaning:

- **Permit** This means that IPsec should be applied to the matching traffic.
- **Deny** This means that the packet should be forwarded and IPsec not applied.

The following access list entry on PIX1 will match all IP traffic from the inside network (192.168.2.0/24), leaving the outside interface to be tunneled to PIX2 (192.168.3.0/24) and the return tunneled IP traffic from 192.168.3.0/24 to 192.168.2.0/24:

```
access-list crypto1 permit ip 192.168.2.0 255.255.255.0 192.168.3.0 255
    .255.255.0
```

A packet from 192.168.2.3 to 192.168.3.4 will be matched by access list *crypto1* and submitted to the IPsec engine. A packet from 192.168.2.3 to www .cisco.com will not be matched and thus transmitted in the clear. Similarly, with return traffic, if an IPsec packet arrives and after decapsulation, it happens to be from 192.168.3.4 to 192.168.2.3, it will be matched by the same access list and forwarded to 192.168.2.3. If the IPsec packet originates from www.cisco.com, it will not be matched and therefore will be dropped. Any clear-text packets from www.cisco.com will pass through and be permitted unmatched.

When the first *permit* entry in an access list is matched, this entry will define the scope of SA that will be created for its protection. For example, in our case all traffic from network 192.168.2.0/24 to the network 192.168.3.0/24 will be protected by the same SA. Let's assume that you create an access list on PIX1 using the following command set:

```
access-list crypto2 permit ip 192.168.2.0 255.255.255.128 192.168.3.0 255
    .255.255.0
access-list crypto2 permit ip 192.168.2.128 255.255.255.128 192.168.3.0
    255.255.255.0
```

In this case, the traffic originating from 192.168.2.0/25 and the traffic from 192.168.2.128/25 will be protected by two different IPsec SAs.

Let's now return to our earlier example and configure the firewalls with access lists:

```
PIX1(config)# access-list crypto1 permit ip 192.168.2.0 255.255.255.0 192
    .168.3.0 255.255.255.0
PIX2(config)# access-list crypto2 permit ip 192.168.3.0 255.255.255.0 192
    .168.2.0 255.255.255.0
```

We are not applying these lists yet. This will be done later using a *crypto map* command.

NOTE

Source addresses in crypto access lists should be the same as they appear on the firewall's outside interface. For example, if NAT is used for translating some of the internal addresses, the global IP addresses must be stated as the access list source, not the local IP addresses. For example, let's assume that the host 192.168.2.25 on the inside interface of PIX1 is translated to 12.23.34.55 on the outside by the following command:

```
static (inside, outside) 12.23.34.55 192.168.2.25 netmask 255.255.
    255.255 0 0
```

In this case, an access list entry for allowing IPsec for this host only should look like:

```
access list crypto1 permit ip host 12.23.34.55 192.168.3.0 255.
    255.255.0
```

Defining a Transform Set

A *transform set* is a set of parameters for a specific IPsec connection (for an IPsec SA, to be precise). It specifies the algorithms used for AH and ESP protocols and the mode (tunnel or transport) in which they are applied. It is possible to configure many different transform sets, but there must be one set shared by both gateways for each crypto map entry so that they can agree on a common set of parameters. Transform sets are configured using the following command:

```
crypto ipsec transform-set <transform-set-name> <transform1> [<transform2>
    [<transform3>]]
```

On the PIX firewall, the default is to use tunnel mode. Transport mode is available only when using the L2TP protocol and is configured using the following command:

```
crypto ipsec transform-set <transform-set-name> mode transport
```

It is possible to configure up to three transforms in a single set: zero or one AH transforms and zero, one, or two ESP transforms. When two ESP transforms are configured, one of them must be an encrypted transform and the other an authentication transform. The available transforms are:

- **ah-md5-hmac** The MD5-HMAC authentication algorithm is chosen for AH.

- **ah-sha-hmac** The SHA-1-HMAC authentication algorithm is chosen for AH.
- **esp-des** The DES encryption algorithm (56-bit key) is chosen for ESP encryption.
- **esp-3des** The Triple DES encryption algorithm (168-bit key) is chosen for ESP encryption.
- **esp-md5-hmac** The MD5-HMAC authentication algorithm is chosen for ESP.
- **esp-sha-hmac** The SHA-1-HMAC authentication algorithm is chosen for ESP.

In our example, we use ESP encryption with DES and authentication with SHA-1-HMAC without AH:

```
PIX1(config)# crypto ipsec transform-set myset esp-des esp-sha-hmac
PIX2(config)# crypto ipsec transform-set myset esp-des esp-sha-hmac
```

Configured transform sets can be checked using the *show crypto ipsec transform-set* command:

```
PIX1(config)# show crypto ipsec transform-set
Transform set myset: { esp-des esp-sha-hmac }
will negotiate = { Tunnel, },
```

Bypassing Network Address Translation

Suppose we use NAT on all outbound traffic from inside networks to the Internet. Because we want to tunnel all traffic between the inside networks on each firewall, we must exclude this traffic from network address translation. To bypass NAT, we can use the *nat 0* command with the same access list that defines our IPsec traffic:

```
PIX1(config)# nat 0 access-list crypto1
PIX1(config)# nat (inside) 1 0 0
PIX1(config)# global (outside) 1 12.23.34.46
PIX2(config)# nat 0 access-list crypto2
PIX2(config)# nat (inside) 1 0 0
PIX2(config)# global (outside) 1 23.34.45.57
```

Configuring a Crypto Map

A *crypto map* connects all other IPsec-related bits together and creates an SPD for a specific interface, through which IPsec traffic is tunneled. A crypto map is identified by its name. An interface can have only one crypto map assigned to it, although this map may have many different entries, identified by their sequence numbers. Entries in a crypto map are evaluated in ascending order. Various entries are equivalent to the various policies in SPD. The first entry that matches the traffic will define methods of its protection. A crypto map entry for IPsec with IKE is created using the following command:

```
crypto map <name> <seq-num> [ipsec-isakmp]
```

The keyword *ipsec-isakmp* is the default and can be omitted. In our example, we create the following entries:

```
PIX1(config)# crypto map pix1map 10 ipsec-isakmp
PIX2(config)# crypto map pix2map 10 ipsec-isakmp
```

Next, specify the traffic selectors for these entries using the command:

```
crypto map <map-name> <seq-num> match address <access-list-name>
```

In our case, these would look like:

```
PIX1(config)# crypto map pix1map 10 match address crypto1
PIX2(config)# crypto map pix2map 10 match address crypto2
```

Now we need to specify the IPsec peers with which the traffic protected by this entry can be exchanged:

```
crypto map <map-name> <seq-num> set peer {<hostname> | <ip-address>}
```

IPsec peers are identified either by their IP addresses or by their hostnames. It is possible to specify multiple peers by repeating this command for one crypto map entry. For our example, we use the following configuration:

```
PIX1(config)# crypto map pix1map 10 set peer 23.34.45.56
PIX2(config)# crypto map pix2map 10 set peer 12.23.34.45
```

Now we need to specify which transform sets can be negotiated for the traffic matching this entry. Multiple (up to six) previously defined transform sets can be specified here:

```
crypto map <map-name> <seq-num> set transform set <transform-set-name1>
    [<transform-set-name2> [<transform-set-name3> [<transform-set-name4>
        [<transform-set-name5> [<transform-set-name6>]]]]]
```

In order for two peers to establish an IPsec tunnel under this crypto map entry, at least one transform set in each firewall's corresponding crypto map entry must have the protocols and encryption/data authentication algorithms. For our simple example, we simply use one transform set on each firewall (*pix1map* on PIX1 and *pix2map* on PIX2):

```
PIX1(config)# crypto map pix1map 10 set transform-set myset
PIX2(config)# crypto map pix2map 10 set transform-set myset
```

In each case, *myset* is the transform set defined previously. It does not need to have the same name on each firewall, but the parameters must match.

The next two steps are optional: requesting that PFS should be used and selecting the SA lifetime. PFS is requested for a crypto map entry using the following command:

```
crypto map <map-name> <seq-num> set pfs [group1 | group2]
```

The *group1* and *group2* keywords denote the DH group and are used for key exchange each time new keys are generated. In order to be effective, PFS has to be configured on both sides of the tunnel; otherwise, if only one peer supports PFS, the IPsec SA will not be established. We will not use this feature in our example.

It is possible to configure a nondefault IPsec SA lifetime for the specific crypto map entry using the following:

```
crypto map <map-name> <seq-num> set security-association lifetime {seconds
    <seconds> | kilobytes <kilobytes>}
```

This command sets a limit on the amount of time an IPsec SA can be used or the maximum amount of traffic that can be transferred by this SA. Right before a timeout or the maximum amount of traffic is reached, the IPsec SA for this crypto map entry is renegotiated. The renegotiations start 30 seconds before a timeout expires or when the volume of traffic is 256KB less than the specified volume lifetime. During this negotiation, one peer sends a proposal to the other, with one of its parameters being an SA lifetime. The second peer selects the lesser of the proposed values and its own lifetime value and sets this as a common SA lifetime.

It is possible to change the default global IPsec SA lifetime using the following command, which has the same parameters:

```
crypto ipsec security-association lifetime {seconds <seconds> | kilobytes
    <kilobytes>}
```

If not specified, the defaults are 28,800 seconds and 4,608,000KB.

The last configuration step is to apply the created crypto map to an interface. The command for doing this is:

```
crypto map <map-name> interface <interface-name>
```

In our case, this will be:

```
PIX1(config)# crypto map pix1map interface outside
PIX2(config)# crypto map pix2map interface outside
```

You can check crypto map configuration using the following command:

```
PIX1(config)# show crypto map
Crypto Map: "pix1map" interface: "outside" local address: 12.23.34.45
Crypto Map "pix1map" 10 ipsec-isakmp
  Peer = 23.34.45.56
  access-list crypto1 permit ip 192.168.2.0 255.255.255.0 192.168.3.0 255
      .255.255.0 (hitcnt=0)
  Current peer: 23.34.45.56
  Security association lifetime: 4608000 kilobytes/28800 seconds
  PFS (Y/N): N
  Transform sets={ myset, }
```

The state of established IPsec SAs can be checked with the *show crypto ipsec sa* command*:*

```
PIX1(config)# show crypto ipsec sa
interface: outside
    Crypto map tag: pix1map, local addr. 12.23.34.45
local  ident (addr/mask/prot/port): (192.168.2.0/255.255.255.0/0/0)
   remote ident (addr/mask/prot/port): (192.168.3.0/255.255.255.0/0/0)
   current_peer: 23.34.45.56
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 10, #pkts encrypt: 10, #pkts digest 0
    #pkts decaps: 12, #pkts decrypt: 17, #pkts verify 0
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0, #pkts decompress
        failed: 0
    #send errors 2, #recv errors 0
```

Troubleshooting

You can debug IKE SA or IPsec SA establishment using the *debug isakmp* and *debug ipsec* commands. These commands tend to produce a lot of output, but they are easy to understand if you know how IPsec works. For example, the following part of a log tells us that IKE negotiations were completed successfully:

```
ISAKMP (0): Checking ISAKMP transform 1 against priority 9 policy
ISAKMP:       encryption DES-CBC
ISAKMP:       hash SHA
ISAKMP:       default group 1
ISAKMP:       auth pre-share
ISAKMP:       life type in seconds
ISAKMP:       life duration (VPI) of  0x0 0x1 0x51 0x80
ISAKMP (0): atts are acceptable. Next payload is 0
ISAKMP (0): SA is doing pre-shared key authentication using id type
     ID_IPV4_ADDR
return status is IKMP_NO_ERROR
```

On the other hand, something similar to the following output will tell you that the IKE main mode exchange failed (IKMP_NO_ERROR_NO_TRANS) because a common proposal (transform set) was not found:

```
VPN Peer: ISAKMP: Added new peer: ip:PIX2 Total VPN Peers:3
VPN Peer: ISAKMP: Peer ip:PIX2 Ref cnt incremented to:1 Total VPN
Peers:3
ISAKMP (0): beginning Main Mode exchange
crypto_isakmp_process_block: src PIX2, dest PIX1
return status is IKMP_NO_ERR_NO_TRANS
ISAKMP (0): retransmitting phase 1...
```

Configuring Site-to-Site IPsec Without IKE (Manual IPsec)

As was described at the beginning of this chapter, IPsec can work without IKE. In this case, all IPsec SAs are established manually. This configuration is more difficult to scale and requires knowledge of the IP addresses or DNS names of all peers. There is also no possibility of reestablishing the SAs, and there is no SA

lifetime configuration or PFS. The main configuration differences with pre-shared key IKE, for example, are:

- No IKE configuration is involved.
- When creating a crypto map entry, specify *ipsec-manual* instead of *ipsec-isakmp.*
- The crypto map configuration must specify encryption and/or authentication keys used for ESP and AH for each tunnel.

Let's briefly go through configuration for a manual IPsec tunnel between PIX1 and PIX2. We assume that all previous configuration of IPsec is deleted on both firewalls. The first few steps are the same (permitting IPsec traffic, defining crypto access lists, creating transform sets, and enabling NAT bypass):

```
PIX1(config)# sysopt connection permit-ipsec
PIX1(config)# access-list crypto1 permit ip 192.168.2.0 255.255.255.0 192
    .168.3.0 255.255.255.0
PIX1(config)# crypto ipsec transform-set myset esp-des esp-sha-hmac
PIX1(config)# nat 0 access-list crypto1
PIX1(config)# nat (inside) 1 0.0.0.0 0.0.0.0
PIX1(config)# global (outside) 1 12.23.34.46
PIX2(config)# sysopt connection permit-ipsec
PIX2(config)# access-list crypto2 permit ip 192.168.3.0 255.255.255.0 192
    .168.2.0 255.255.255.0
PIX2(config)# crypto ipsec transform-set myset esp-des esp-sha-hmac
PIX2(config)# nat 0 access-list crypto2
PIX2(config)# nat (inside) 1 0.0.0.0 0.0.0.0
PIX2(config)# global (outside) 1 23.34.45.57
```

The next step is to create the crypto maps. The following commands specify that manually configured IPsec SAs will be used.

```
PIX1(config)# crypto map pix1map 10 ipsec-manual
PIX2(config)# crypto map pix1map 10 ipsec-manual
```

The rest of the crypto map configuration is the same as with IKE:

```
PIX1(config)# crypto map pix1map 10 match address crypto1
PIX1(config)# crypto map pix1map 10 set peer 23.34.45.56
PIX1(config)# crypto map pix1map 10 set transform-set myset
PIX2(config)# crypto map pix2map 10 match address crypto2
```

```
PIX2(config)# crypto map pix2map 10 set peer 12.23.34.45
PIX2(config)# crypto map pix2map 10 set transform-set myset
```

Now we manually need to define the configuration of the SAs for each transform that is used. We had ESP with encryption and ESP with authentication in the transform set *myset*, so we need to specify two outbound SAs and two inbound SAs. (Remember, each SA exists for one transform and in one direction.) The PIX makes this process a little easy, allowing the configuration of both the encryption and the authentication keys in one command. However, specification of data for inbound and outbound traffic must still be defined separately. We will use the following command:

```
crypto map <map-name> <seq-num> set session-key inbound | outbound esp
    <spi> cipher <hex-key-string> [authenticator <hex-key-string>]
```

The *map-name* and *seq-num* parameters have been discussed before. The *spi* parameter is a numerical value of the Security Parameter Index. This number is arbitrary, although it has one requirement that an SPI number for, say, IPsec SA, which is responsible for ESP protection of outbound traffic on one peer, has to be the same as the SPI for the IPsec SA responsible for ESP protection of inbound traffic on the second peer. This holds true with the keys (*hex-key-string*); the key for an outbound SA on one peer has to be the same as the key for the corresponding inbound SA on the second peer. The key value can be 16, 32, or 40 hexadecimal digits. There are some minimal requirements on key length:

- If a transform set for this map entry includes DES encryption, specify at least a 16-digit key.
- If this transform set includes the MD5 algorithm, specify at least 32 digits per key.
- If it includes the SHA-1 algorithm, specify at least 40 digits per key.

If a longer key is specified, it is simply hashed (not truncated) to the required length. For PIX1, we will specify the following SPIs and keys:

```
PIX1(config)# crypto map pix1map 10 set session-key inbound esp 300 cipher
    1234455667788909 authenticator 123445566778890acdefacd91234455667788909
PIX1(config)# crypto map pix1map 10 set session-key outbound esp 400 cipher
    9887766554344556 authenticator acdefacd12238474646537485956745637485635
```

They include a 16-digit DES key and a 40-digit SHA-1 key.

On the second firewall we have to create a "mirror" configuration of keys and SPIs, applying the same commands but with *inbound* and *outbound* interchanged:

```
PIX2(config)# crypto map pix2map 10 set session-key outbound esp 300
    cipher 1234455667788909 authenticator 123445566778890acdefacd91234
        455667788909
PIX2(config)# crypto map pix2map 10 set session-key inbound esp 400
    cipher 9887766554344556 authenticator acdefacd1223847464653748595674
        5637485635
```

If we were using AH for traffic authentication, we would add the following command twice (one for the inbound and one for the outbound IPsec SA) to the configuration of each firewall:

```
crypto map <map-name> <seq-num> set session-key outbound ah <spi>
    <hex-key-data>
```

This uses the same agreements but requires only one key for each SPI. After applying the crypto map to the outside interfaces on both firewalls, the configuration is complete:

```
PIX1(config)# crypto map pix1map interface outside
PIX2(config)# crypto map pix2map interface outside
```

Configuring Point-to-Point Tunneling Protocol

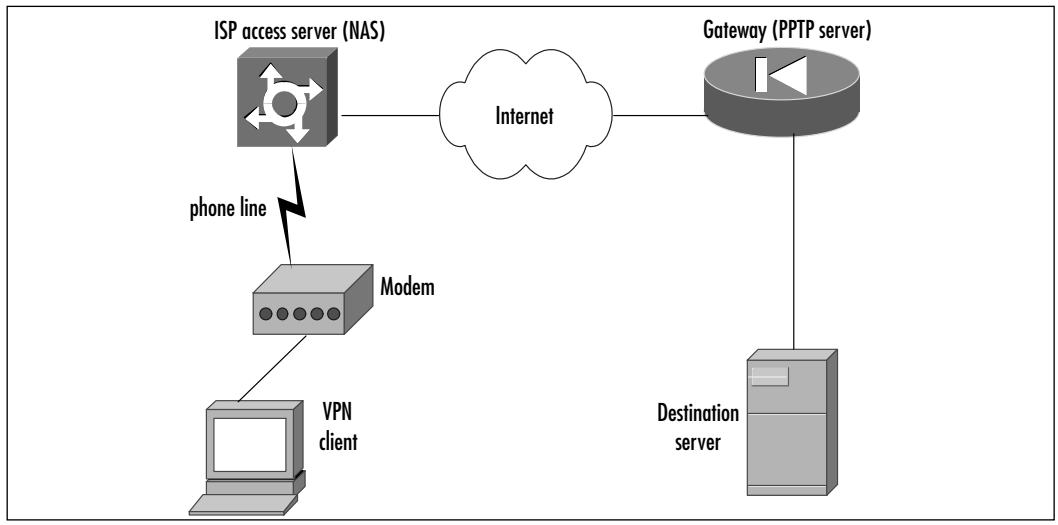
Point-to-Point Tunneling Protocol (PPTP), defined in RFC 2637, is another common protocol used for establishing VPNs. The biggest difference between IPsec and PPTP is that while IPsec is focused on tunneling IP traffic, PPTP works at Layer 2 and has the ability to tunnel any Layer 3 traffic, including non-IP protocols. Although PPTP is usually associated with Microsoft (the Windows OS has included PPTP client and server functionality since NT 4.0), it was actually designed by the PPTP Forum, which includes Microsoft Corporation, Ascend Communications, 3Com/Primary Access, ECI Telematics, and US Robotics.

Overview

PPTP is much simpler than IPsec in its structure (see Figure 7.9). Each tunnel includes the following elements:

- The client
- A network access server (for example, an ISP's dialup server)
- The gateway or PPTP server

Figure 7.9 Point-to-Point Tunneling Protocol Functionality



When a connection is being established, the following happens:

1. A client connects to the public network (establishes a dialup connection with an ISP, for example). If a connection is established, this step is optional.
2. A PPTP control connection (a connection from the client to TCP port 1723 on the server) is established. This connection is known as the *PPTP tunnel*.
3. A General Routing Encapsulation (GRE) tunnel is established over IP 47 and is known as the PPTP data tunnel.
4. All Layer 3 protocols are encapsulated by the client into PPP packets first and then transmitted through the GRE tunnel. This traffic is decapsulated twice (from GRE and from PPP) on the other side by the gateway and then forwarded to the private network.

NOTE

When using PPTP, be sure to check that no network devices between client and gateway (for example, ISP routers) filter IP 47 (GRE) and TCP connections to port 1723 on the gateway (the PIX firewall in our case).

The PIX firewall supports inbound PPTP. It can function as a server but not as a client. Another restriction is that only one of its interfaces can have PPTP processing enabled.

As PPTP is PPP encapsulated into GRE, it uses all PPP authentication and encryption features. Authentication here means client authentication only (using PAP, CHAP, or MS-CHAP), as opposed to IPsec packet authentication. Unfortunately, PPTP allows packet spoofing and insertion by third parties, but this threat can be eliminated to a certain degree by using encryption. Authentication can be performed by the PIX firewall using either its internal database or external AAA servers (RADIUS or TACACS+).

Encryption is negotiated using PPP Compression Control Protocol (CCP). One of the available options in this protocol is the encryption bit. When it is turned on, the tunneled PPP connection uses RC4 encryption with 40-bit or 128-bit keys—a part of Microsoft Point-to Point Encryption (MPPE) extensions. As with DES, longer keys are recommended, especially since RC4 is even weaker than DES. Compression itself is not supported in PIX version 6.2. When MPPE is used, the external AAA server used for authentication must be RADIUS, and it should be able to return a MSCHAP_MPPE_KEY attribute to the PIX firewall in the RADIUS Authentication Accept packet. This Microsoft-specific RADIUS attribute is described (among others) in RFC 2548.

NOTE

MMPE can be used only if MS-CHAP authentication is supported, because MMPE needs an initial key to be generated during authentication process, and this is possible only with MS-CHAP.

The PIX uses another PPP subprotocol, IP Control Protocol (IPCP), to assign an internal IP address from the specified PPTP pool to the client. The PIX firewall only supports 255 concurrent PPTP client connections.

Configuration

In this section, we will configure the PIX firewall to accept PPTP connections. Most of the PPTP configuration tasks on the PIX are performed using *vpdn* commands. *VPDN* stands for *Virtual Private Dialup Networking* and is used on the PIX as a common term for PPTP, L2TP, and PPPoE configurations. As with IPsec, the first step is to permit incoming PPTP traffic. This is done using the following command:

```
sysopt connection permit pptp
```

This command implicitly allows all traffic from authenticated PPTP clients to pass to its destination without additional conduits or access lists. Without this command it is required to create additional entries in the access lists on the outside interface, because even if dial-in clients obtain internal IP addresses, their packets still arrive on the outside interface.

The rest of configuration consists of the following:

1. Creating an address pool for PPTP clients
2. Creating an AAA scheme if external AAA servers are used
3. Creating a dial-in group (VPDN group) and setting dial-in parameters such as authentication and encryption
4. Creating access lists, which allows PPTP clients to access internal servers (if you did not specify the *sysopt connection permit pptp* command)

An IP address pool is created using the following command:

```
ip local pool <pool_name> <pool_start_address>[-<pool_end_address>]
```

This command creates a named pool (*pool_name* can be any alphanumeric name) with the starting address *pool_start_address* and the ending address *pool_end_address*. For example:

```
PIX1(config)# ip local pool mypool 10.1.1.1-10.1.1.10
```

This command allocates 10 IP addresses to the pool of available addresses. The state of this pool can be displayed using the *show ip local pool <pool_name>* command:

```
PIX1# show ip local pool mypool
Pool      Begin        End          Free      In use
mypool    10.0.1.1     10.0.0.10    10        0
```

```
Available Addresses:
10.0.1.1
10.0.1.2
10.0.1.3
10.0.1.4
10.0.1.5
10.0.1.6
10.0.1.7
10.0.1.8
10.0.1.9
10.0.1.10
```

When all the IP addresses from the pool are allocated and a new allocation attempt fails, the PIX creates a syslog message of the type:

```
%PIX-3-213004: PPP virtual interface number client ip allocation failed.
```

Let's assume for a moment that we will not be using external AAA servers. (The case of external authentication is described later.) We have to configure a series of local usernames and passwords. This is done using the following command:

```
vpdn username <name> password <pass>
```

For example:

```
PIX1(config)# vpdn username user1 password password1
PIX1(config)# vpdn username user2 password password2
```

These two commands create two users, *user1* with password *password1* and *user2* with password *password2*. The next step is to create a VPDN group. The minimal configuration without any authentication requires three commands:

```
vpdn group <group_name> accept dialin pptp
vpdn group <group_name> client configuration address local <address_pool
    _name>
vpdn enable <interface>
```

The first command enables processing of PPTP traffic by the group. The second specifies the IP address pool to be used for clients. The third command applies VPDN settings to the interface (usually an outside interface). If local authentication is used, the following commands are added:

```
vpdn group <group_name> ppp authentication {pap | chap| mschap}
vpdn group <group_name> client authentication local
```

The first command selects the authentication mode (PAP, CHAP, or MS-CHAP). The PIX supports only MS-CHAP version 1, not 2. In all cases, the same authentication protocol should be configured on PIX and on the dial-in client. If this command is not present in the PIX configuration, no authentication is performed and any client is allowed. The second line specifies that a local database will be used for authentication. When an external AAA server is used, this server is configured by usual AAA means. For example:

```
PIX1(config)# aaa-server myserver (inside) host 192.168.2.99 key
    mysecretkey
PIX1(config)# aaa-server myserver protocol radius
```

This server is then specified in a VPDN group using the following command:

```
vpdn group <group_name> client authentication aaa <aaa-server-group>
```

In our case, this will be:

```
PIX1(config)# vpdn group mygroup client authentication aaa myserver
```

Encryption is specified by the following command:

```
vpdn group <group_name ppp> encryption mppe 40 | 128 | auto [ required ]
```

Here, 40, 128, or "auto" specifies the length of the encryption key. Again, it must match client settings. The *auto* keyword means that the PIX will accept both 40- and 128-bit keys. The *required* keyword means that if the client refuses to support encryption with the key of specified length, the connection will be dropped.

NOTE

If the PIX requires a 128-bit encryption key but Windows 95/98 client supports only 40-bit encryption (older exported versions), the initial connection appears to be accepted; Windows moves a connection icon to the taskbar, but PPP option negotiation is still in progress. The PIX will refuse the PPTP tunnel; a Windows client will not be disconnected immediately but will be eventually timed out.

It is possible to specify DNS and WINS server settings to be passed on to the client with the following commands:

```
vpdn group <group_name> client configuration dns <dns_server1> [<dns_
    server2>]
vpdn group <group_name> client configuration wins <wins_server1> [<wins_
    server2>]
```

Let's consider some examples of PPTP configuration. The following is a configuration with local MS-CHAP authentication and no encryption:

```
ip local pool mypool 192.168.3.1-192.168.3.10
vpdn username user1 password password1
vpdn username user2 password password2
vpdn group 1 accept dialin pptp
vpdn group 1 ppp authentication mschap
vpdn group 1 client authentication local
vpdn group 1 client configuration address local mypool
vpdn enable outside
sysopt connection permit pptp
```

If we need more granular access to internal servers, we can replace the *sysopt* command from the preceding listing with an access list on the outside interface. For example, to allow PPTP clients to access only Telnet service to the internal host 192.168.2.33, which has an outside address 12.23.34.99, the following configuration can be used:

```
ip local pool mypool 192.168.3.1-192.168.3.10
vpdn username user1 password password1
vpdn username user2 password password2
vpdn group 1 accept dialin pptp
vpdn group 1 ppp authentication mschap
vpdn group 1 client authentication local
vpdn group 1 client configuration address local mypool
vpdn enable outside
static (inside, outside) 12.23.34.99 192.168.2.33
access-list acl_out permit tcp 192.168.3.0 255.255.255.240 host 12.23.34
    .99 eq telnet
access-group acl_out in interface outside
```

Note that when the *sysopt connection permit pptp* command is absent, decapsulated PPTP traffic is subject to all rules and access lists applied to the inbound traffic.

Here is a more complex example in which clients authenticate with MS-CHAP version 1 via an external RADIUS server, 128-bit encryption is required, and clients receive DNS and WINS settings from the PIX:

```
ip local pool mypool 192.168.3.1-192.168.3.10
aaa-server myserver (inside) host 192.168.2.99 key mysecretkey
aaa-server myserver protocol radius
vpdn group 1 accept dialin pptp
vpdn group 1 ppp authentication mschap
vpdn group 1 client authentication aaa myserver
vpdn group 1 ppp encryption mppe auto required
vpdn group 1 client configuration address local mypool
vpdn group 1 client configuration dns 192.168.2.33
vpdn group 1 client configuration wins 192.168.2.34
vpdn enable outside
sysopt connection permit pptp
```

The status of PPTP tunnels can be displayed using several commands:

```
PIX1# show vpdn tunnel
% No active L2TP tunnels
% No active PPTP tunnels
```

If any tunnels were active, statistics on their number and traffic would have been displayed:

```
PIX1# show vpdn tunnel pptp packet
PPTP Tunnel Information (Total tunnels=1 sessions=1)
LocID    Pkts-In     Pkts-Out       Bytes-In       Bytes-Out
1         1234         23             200323          553
```

The preceding command shows only the traffic statistics for active PPTP data tunnels. Another command is used to monitor PPTP tunnels themselves:

```
PIX1# show vpdn tunnel pptp summary
PPTP Tunnel Information (Total tunnels=1 sessions=1)
LocID      RemID      State      Remote Address       Port Sessions
1            1          estabd     172.16.38.194        1723 1
```

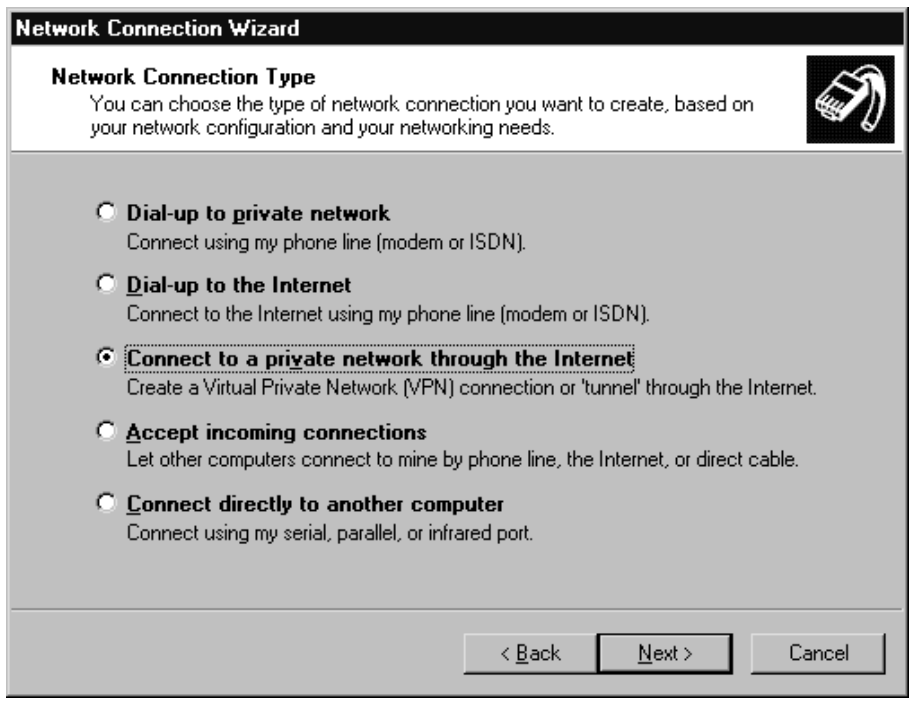
The following commands display transport layer statistics and session information, respectively:

```
show vpdn tunnel pptp transport
show vpdn pptp session
```

Setting Up Windows 2000 Clients

Client setup for MS Windows 2000 is rather simple and can be done using the Make New Connection wizard, located in the **Start** menu under **Settings | Network and Dialup Connections**. Click **Next** in the "Welcome to the network connection wizard" screen and select **Connect to a private network through the Internet** in the **Connection type** (see Figure 7.10).

Figure 7.10 Setting Up Windows 2000 VPN Clients



The next screen, "Public Network," asks you to select an ISP connection to be dialed before the VPN tunnel is established (a NAS connection, in terms of Figure 7.11). If you have a permanent connection to the Internet, you have to select **Do not dial the initial connection**.

In the following screen, you need to enter the IP address of PIX firewall's outside interface (the interface on which PPTP connections are accepted). See Figure 7.12.

The next two screens ask you to select local users who can use this connection and to enter the name for this connection. After the wizard finishes, a new connection icon appears (see Figure 7.13).

Figure 7.11 Selecting a Dialup Connection

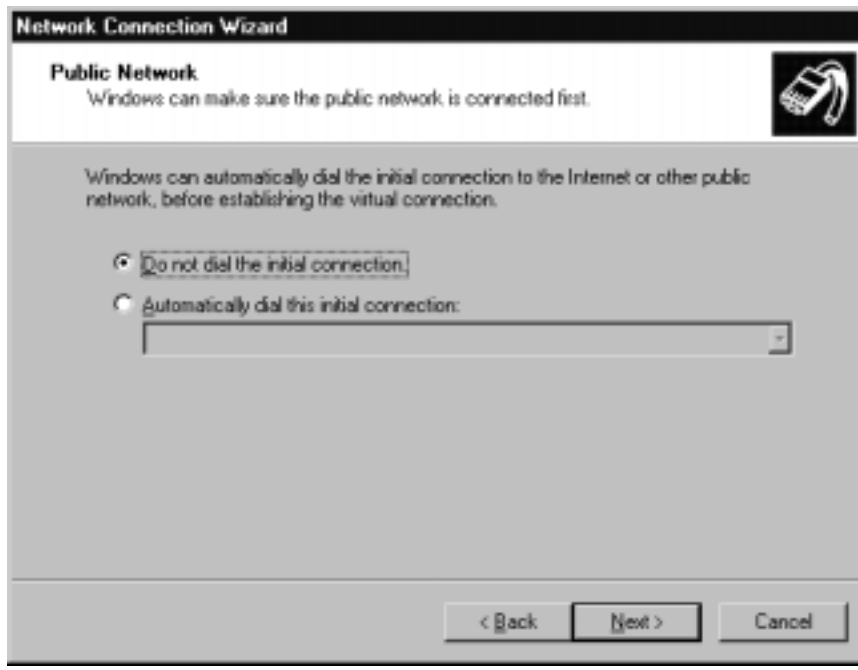


Figure 7.12 Gateway Address

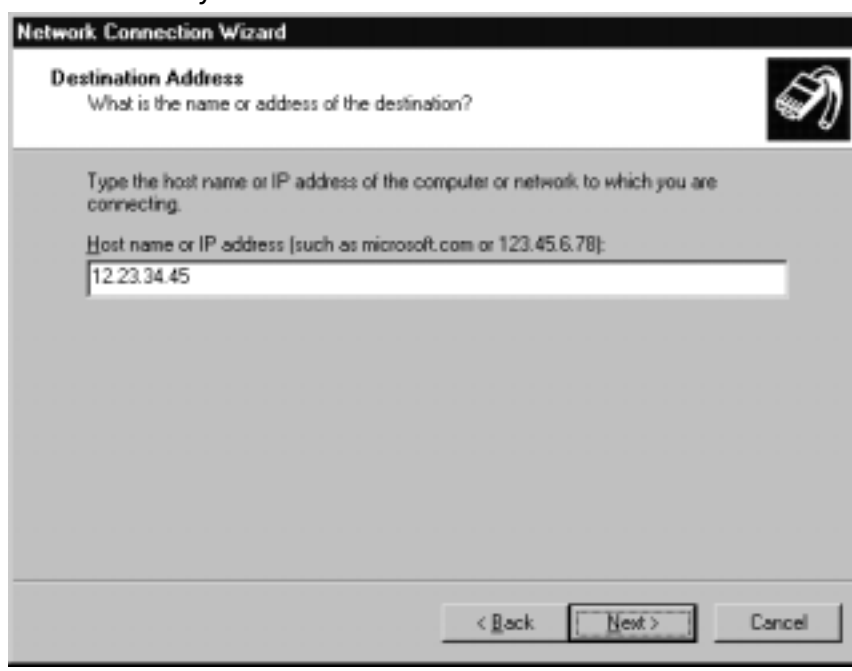
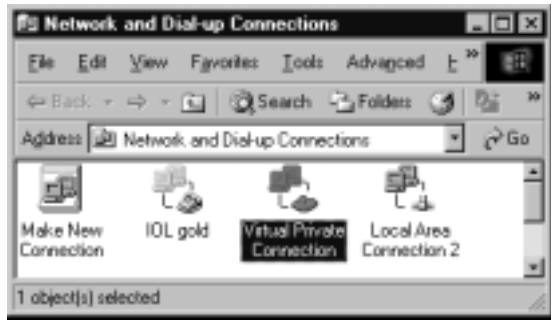
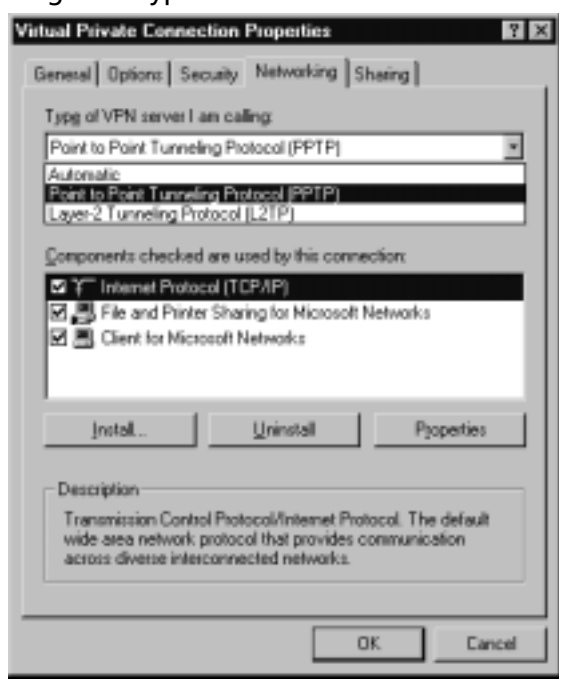


Figure 7.13 Network Connections with One VPN Link Created



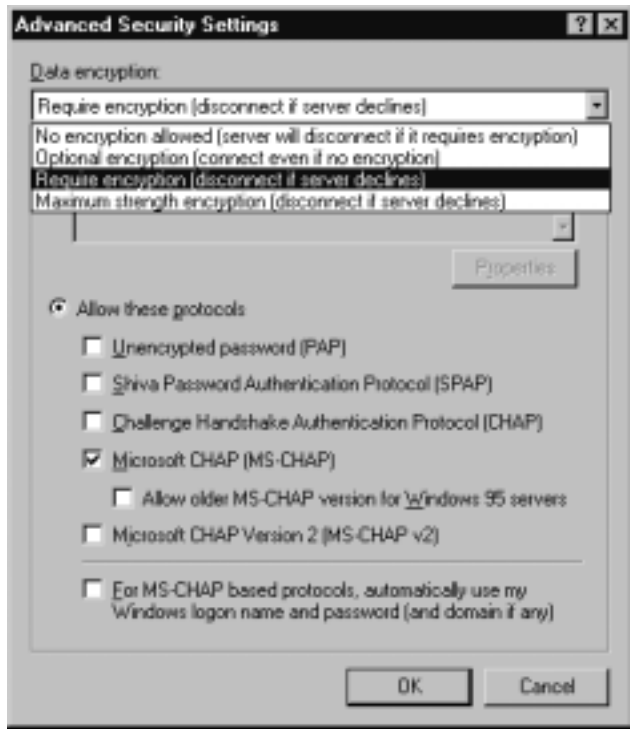
You need to check some parameters before you can use this link. If you open the **Properties** screen for this connection, you'll see that the **General** tab contains information about the dialup connection and the PPTP gateway. The **Options** and **Sharing** tabs are the same as dialup connections. The **Networking** tab needs some tweaking. First, it is recommended that a specific VPN type (PPTP) be selected instead of the default **Automatic** setting (see Figure 7.14). Also, it is recommended that only the protocols that will be used on this connection are selected (for example, TCP/IP).

Figure 7.14 Selecting VPN Type and Tunneled Protocols



On the **Security** tab, select **Advanced** and click **Settings**. On the following tab (see Figure 7.15), you need to select the settings to match the PIX VPDN encryption and authentication settings.

Figure 7.15 Encryption and Authentication



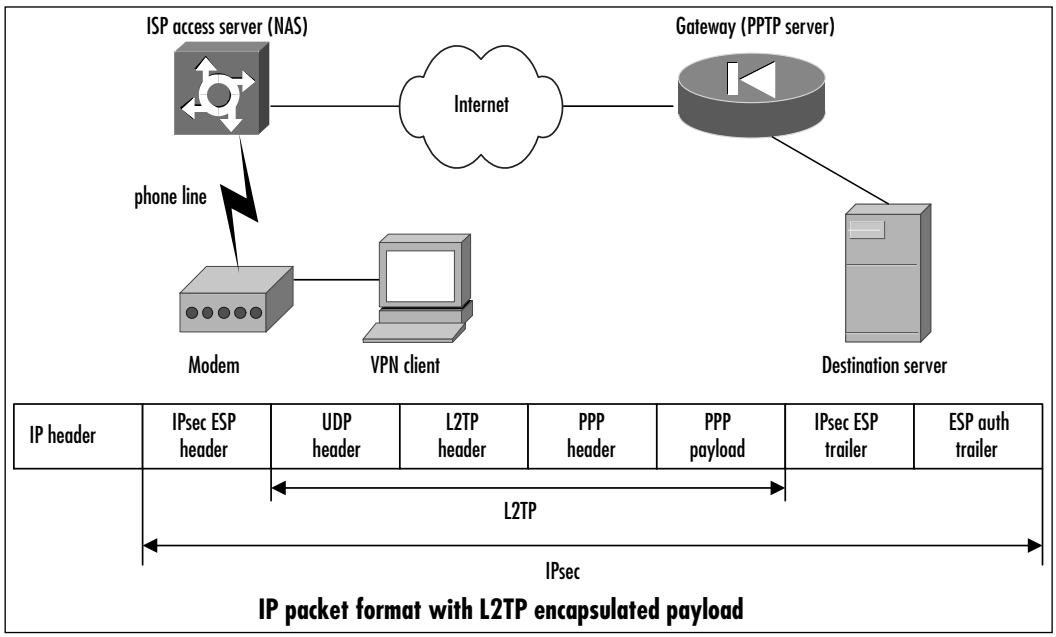
Most of the settings here are self-explanatory. *Maximum strength encryption* here means that the client will agree to use only 128-bit encryption and will disconnect if the PIX is configured to use only 40-bit keys. When the PIX is configured to use MS-CHAP authentication, you need to check only one box (as shown) because the PIX supports only version 1 of this authentication protocol.

Configuring Layer 2 Tunneling Protocol with IPsec

Another protocol for tunneling all Layer 2 traffic over public network is Layer 2 Tunneling Protocol (L2TP). L2TP is a hybrid of Cisco's Layer 2 Forwarding Protocol (L2F) and PPTP. In this section, you will learn how to configure L2TP with IPsec. L2TP/IPsec works as follows: The parties first establish an IPsec tunnel in transport mode using IKE or manual configuration, and then the traffic

between the networks is encapsulated in PPP packets and transmitted between UDP ports 1701 on both the client and the server through the IPsec tunnel (see Figure 7.16). Thus, configuration consists of two parts: IPsec configuration and VPDN configuration (the latter is very similar to PPTP).

Figure 7.16 Layer 2 Tunneling Protocol Packet Structure



Overview

As shown in Figure 7.16, L2TP support in the PIX is a bit more complicated by its structure, but it is made secure because it allows full IPsec transport mode authentication and encryption of transmitted packets. Since PIX software version 6.0, this implementation can operate with a Windows 2000 client.

Many features of the PIX L2TP server are similar to the PPTP server implementation. L2TP can be configured only on one interface, and it uses PPP authentication methods for client authentication. The PIX cannot serve as an L2TP client.

Dynamic Crypto Maps

One new feature that is used in L2TP configuration is a *dynamic crypto map.* A dynamic crypto map is a crypto map without all parameters configured. It is added as part of the interface's crypto map and is used by the PIX to establish IPsec connections with peers whose IP addresses are not known in advance. A common

example is the case of mobile users; they do not have a predetermined IP address but usually receive a new address each time they dial into their ISP. When the PIX uses dynamic crypto maps, mobile users have to authenticate to the firewall first by something (hostname, for example) during IKE exchange, and then their traffic is processed under the rules defined by the dynamic crypto map entry.

In order to configure a dynamic crypto map entry, you need to specify only a transform set. All other parameters can be accepted from the other peer's (or mobile client, for example) proposals. Dynamic maps can be used only for incoming connections and must be the lowest priority. When the PIX decides to use a specific dynamic map (meaning that it has performed a successful IKE exchange with the peer), it creates a temporary crypto map entry and installs it into its SPD. The entry is filled in with the results of IKE negotiations. Once established, this temporary entry is used as normal. When all IPsec SAs associated with this entry expire, the temporary entry is deleted.

Configuration commands for the dynamic crypto maps are similar to those for static crypto map entries. The configuration commands are as follows:

```
crypto dynamic-map <dynamic-map-name> <dynamic-seq-num>
crypto dynamic-map <dynamic-map-name> <dynamic-seq-num> match address
    <acl_name>
crypto dynamic-map <dynamic-map-name> <dynamic-seq-num> set peer
    {<hostname> | <ip-address>}
crypto dynamic-map <dynamic-map-name> <dynamic-seq-num> set pfs [group1 |
    group2]
crypto dynamic-map <dynamic-map-name> <dynamic-seq-num> set security-
    association lifetime {seconds <seconds> | kilobytes <kilobytes>}
crypto dynamic-map <dynamic-map-name> <dynamic-seq-num> set transform-
    set transform-set-name1 [transform-set-name2 [transform-set-name3
        [transform-set-name4 [transform-set-name5 [transform-set-name6
            [transform-set-name7 [transform-set-name8 [transform-set-
                name9]]]]]]]]
```

Only the last part of the command, the transform set specification, must be present in the configuration of a dynamic crypto map entry. It is also recommended that an access list be specified in the *match address* command to increase security. For example, broadcast and multicast traffic rarely needs to be tunneled.

A configured dynamic crypto map is then assigned as an entry in a regular crypto map. For example:

```
crypto ipsec transform-set myset1 esp-des esp-md5-hmac
crypto ipsec transform-set myset2 ah-sha-hmac
crypto dynamic-map dynmap 10
crypto dynamic-map dynmap set transform-set myset2
crypto dynamic-map dynmap match address 101
crypto dynamic-map dynmap 20
crypto dynamic-map dynmap set transform-set myset1
crypto dynamic-map dynmap match address 102
crypto map gorilla 10 ipsec-isakmp
crypto map gorilla 10 set peer 23.34.45.56
crypto map gorilla 10 set transform-set myset1 myset2
crypto map gorilla 10 match address 103
crypto map gorilla 20 ipsec-isakmp dynamic dynmap
access-list 103 permit ip 192.168.3.0 255.255.255.0 any
access-list 101 permit ip host 192.168.2.33 any
access-list 102 permit ip host 192.168.2.34 any
```

Here, a regular map, *gorilla*, has a dynamic map entry called *dynmap* with priority 20. The dynamic map itself has two entries with priorities 10 and 20. This means that the PIX will first evaluate the static entry with priority 10 (the one with peer 23.34.45.65), and if this entry does not apply, it will try both entries from the dynamic map—the one with priority 10 first, then the entry with priority 20. After an IPsec SA is established, only the traffic specified by the corresponding access list will be tunneled.

Configuration

Configuring L2TP on the PIX for a Windows 2000 client consists of three high-level steps:

1. Configure IKE.
2. Configure IPsec in transport mode.
3. Configure VPDN dial-in settings for L2TP.

IKE is configured as before. Since the internal Windows 200 VPN client does not support pre-shared keys, it has to be configured for CA support. Once the normal configuration procedure for configuring IPsec with CA support has been completed, we can configure PIX1 to allow L2TP VPN connections from mobile Windows 2000 users. In our example, they will be allowed access to an internal

host 192.168.2.33. IKE authentication will be done using VeriSign certificates, and user authentication will be handled by a RADIUS server on the internal network. First, we need to allow IPsec and L2TP traffic to be exempt from conduits. This is done using the following commands:

```
PIX1(config)# sysopt connection permit ipsec
PIX1(config)# sysopt connection permit l2tp
```

CA support is configured the same as before:

```
PIX1(config)# hostname PIX1
PIX1(config)# domain-name securecorp.com
PIX1(config)# ca generate rsa key 1024
PIX1(config)# ca identity verisign 205.139.94.230
PIX1(config)# ca configure verisign ca 1 20 crloptional
PIX1(config)# ca authenticate verisign
PIX1(config)# ca enroll verisign midnightinmoscow
PIX1(config)# ca save all
PIX1(config)# write memory
```

IKE is configured the same as before:

```
PIX1(config)# isakmp policy 10 authentication rsa-sig
PIX1(config)# isakmp policy 10 encryption 3des
PIX1(config)# isakmp policy 10 hash md5
PIX1(config)# isakmp policy 10 group 2
PIX1(config)# isakmp policy 10 lifetime 2400
PIX1(config)# isakmp identity hostname
PIX1(config)# isakmp enable outside
```

NOTE

It is important that the IKE and IPsec SA lifetimes on the PIX match the corresponding settings on the Windows computer. The defaults should work; IKE SA lifetime is 3600 sec and IPsec SA lifetime is 86400 sec on Windows, but if *debug crypto isakmp* or *debug crypto ipsec* indicate that negotiation failed but the transform sets are correct, always check the lifetime settings on both the PIX and the Windows client. Windows settings can be found under the IP security policies snap-in of the Microsoft Management Console. See Microsoft knowledgebase article Q259335 for details.

We should continue the IPsec configuration by defining the crypto access list and configuring NAT bypass:

```
PIX1(config)# access-list 99 permit ip 192.168.2.0 255.255.255.0 any
```

IPsec traffic has to be exempt from the NAT, as it was before:

```
PIX1(config)# nat (inside) 0 access-list 99
```

The next step is to configure the transform set. The only difference from generic IPsec here is that we need to specify that the IPsec mode is *transport*:

```
PIX1(config)# crypto ipsec transform-set myset esp-des esp-md5-hmac
PIX1(config)# crypto ipsec transform-set myset mode transport
```

We create a simple dynamic crypto map to process mobile clients with unspecified IP addresses:

```
PIX1(config)# crypto dynamic-map mobileclients 10 set transform-set myset
PIX1(config)# crypto dynamic-map mobileclients 10 match address 99
```

We configure and apply the regular crypto map, which includes this dynamic map as an entry:

```
PIX1(config)# crypto map partners 20 ipsec-isakmp dynamic mobileclients
PIX1(config)# crypto map partners interface outside
```

IKE and IPsec configuration is now complete. Next we need to configure the VPDN settings. Almost all the commands are identical to PPTP:

```
vpdn group <group_name> accept dialin l2tp
vpdn group <group_name> l2tp tunnel hello <hello_timeout>
vpdn group <group_name> client configuration address local <address_pool_
    name>
vpdn group <group_name> client configuration dns <dns_ip1> [<dns_ip2>]
vpdn group <group_name> client configuration wins <wins_ip1> [<wins_ip2>]
vpdn group <group_name> client authentication aaa <aaa_server_group>
vpdn group <group_name> client authentication local
vpdn group <group_name> ppp authentication {pap | chap | mschap}
vpdn group <group_name> client accounting <aaa_server_group>
```

The first command turns on processing of L2TP requests. The second can be used to configure an L2TP keep-alive timeout, which is 60 seconds by default and can vary from 10 to 300 seconds. The other commands are the same as

PPTP. We will use the last example from the PPTP section (with an external AAA server) and change the configuration to L2TP. The resulting VPDN configuration is as follows:

```
PIX1(config)# ip local pool mypool 192.168.5.1-192.168.5.10
PIX1(config)# aaa-server myserver (inside) host 192.168.2.99 key
    mysecretkey
PIX1(config)# aaa-server myserver protocol radius
PIX1(config)# vpdn group 1 accept dialin l2tp
PIX1(config)# vpdn group 1 ppp authentication mschap
PIX1(config)# vpdn group 1 client authentication aaa myserver
PIX1(config)# vpdn group 1 client configuration address local mypool
PIX1(config)# vpdn group 1 client configuration dns 192.168.2.33
PIX1(config)# vpdn group 1 client configuration wins 192.168.2.34
PIX1(config)# vpdn enable outside
```

Setting Up the Windows 2000 Client

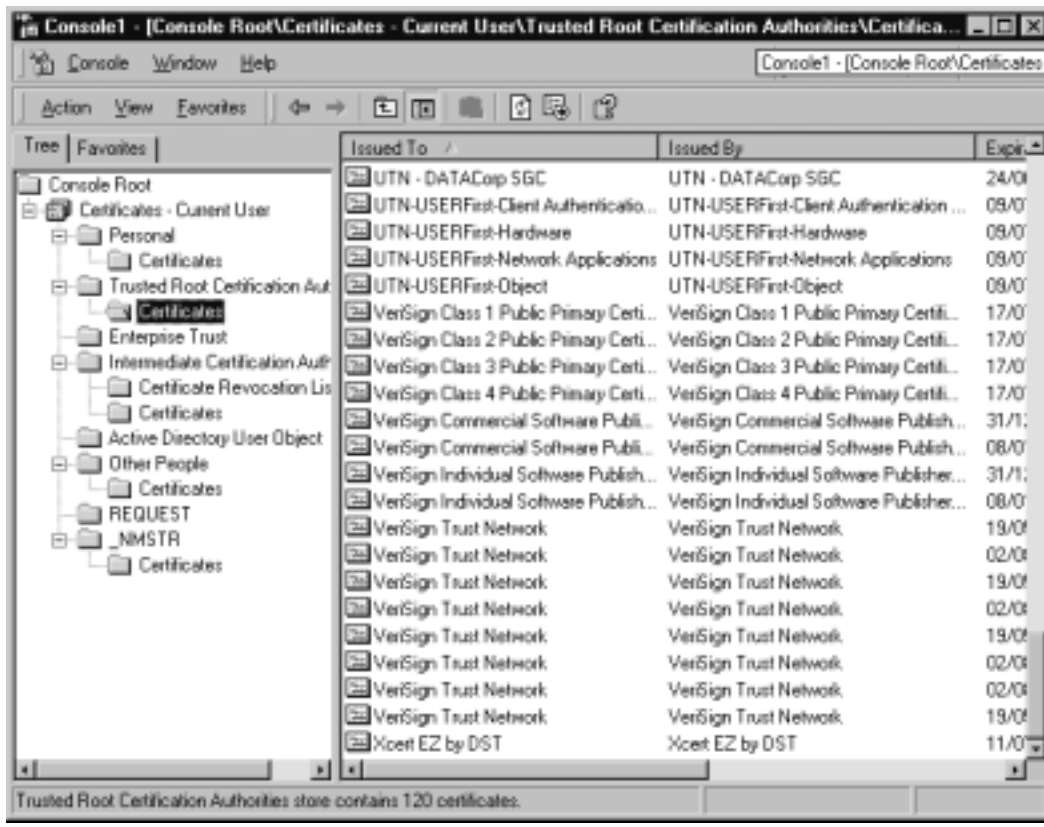
The creation of a VPN connection to be used for L2TP follows the same steps as for PPTP; the only difference is that you need to select L2TP instead of PPTP. Again, authentication settings must match the settings on the PIX firewall.

Additionally, you need to configure certificates and certificate authorities that are used to establish an IPsec tunnel. In Windows 2000, certificate management is performed using the Certificates snap-in of the Microsoft Management Console (use mmc.exe to open C:\winnt\system32\certmgr.msc). An example can be found in Figure 7.17.

You need to check that the certificate of the CA the PIX uses is listed under Trusted Root Certification Authorities. If it is not, you need to obtain it from the CA (the procedure depends on the CA but is usually simple) and import it. You have to do the same with the certificate for your computer (it should be under Personal).

For more details, see Microsoft support article Q253498, *How to Install a Certificate for Use with IP Security*. A good article about L2TP troubleshooting is article Q259335, *Basic L2TP/IPSec Troubleshooting in Windows*.

Figure 7.17 The MMC for Certificate Management



Configuring Support for the Cisco Software VPN Client

The Cisco software VPN client is client software for use with Cisco-based IPsec gateways. It supports Cisco VPN concentrators, PIX, and IOS-based devices. The VPN client is installed on a client computer and takes preference over the internal Windows IPsec client.

NOTE

The internal Windows client will not work when the Cisco software VPN client is installed because it takes over IKE port 500. So, for example, L2TP tunneling described in the previous section will not work.

The latest version of the Cisco VPN client can be downloaded from Cisco's Web site. (You might be required to log in first.) Installation of the Cisco VPN client is straightforward; it might ask you a couple of questions, for example, if you want to remove Internet connection sharing and disable the Windows internal IPsec policy service because the VPN client is not compatible with these two features.

Mode Configuration

IKE mode configuration is an extension of the IKE protocol that allows you to assign a known internal IP address to the VPN client during the IKE negotiation process. The client uses this address later as an "internal" IP address in its communications over the IPsec tunnel. Because this address is already known to the firewall, it can easily be matched against the security policy (SPD). IKE mode configuration allows for easy scalability of VPN networks, which have many clients without fixed IP addresses.

IKE mode configuration occurs between Phases 1 and 2 of IKE negotiation. During this process, it is possible to download an IP address and other IP-related settings such as DNS servers to the client. There are two types of IKE mode configuration negotiation:

- **Gateway initiation** The server initiates the configuration mode with the client. After the client responds, IKE modifies the sender's identity, the message is processed, and the client receives a response.
- **Client initiation** The client initiates the configuration mode with the gateway. The gateway responds with an IP address it has allocated for the client.

There are three steps to configure IKE mode configuration on PIX firewall:

1. Define an IP address pool, as was done, for example, in the section about L2TP. The command is as follows:

```
ip local pool pool_name pool_start_address[-pool_end_address]
```

2. Reference the IP address pool in the IKE configuration using the command:

```
isakmp client configuration address-pool local <pool-name>
    [<interface-name>]
```

This command states that IKE on interface *interface-name* should use the address pool named *pool-name* to assign local IP addresses to VPN clients.

3. In the crypto map settings, define the crypto map settings that should try to negotiate IKE mode configuration with the client and whether the client or gateway will be initiating this process. The relevant command is:

```
crypto map <map-name> client configuration address initiate |
    respond
```

In this command, *map-name* is the name of crypto map and *initiate* means that the gateway initiates IKE mode configuration, and *respond* means that client should start the process itself and the gateway responds. For example:

```
ip local pool modeconf 172.16.1.1-172.16.1.126
isakmp client configuration address-pool local modeconf outside
crypto map mymap client configuration address initiate
```

These settings (if all the rest of IKE and IPsec is configured) will force PIX to try to initiate IKE mode configuration with each client who matches crypto map *mymap*. Clients will be assigned IP addresses from the 172.16.1.1–172.16.1.126 address range.

One slight complication arises if the same interface is used for terminating both VPN clients and peers with static IP addresses (site-to site gateways). Such peers have to be excluded from the IKE mode configuration process. This exclusion is performed using the command:

```
isakmp key <keystring> address <ip-address> [<netmask>] no-config-mode
```

For peers that use pre-shared keys authentication and another command for peers that use RSA signatures use this command:

```
isakmp peer fqdn <fqdn> no-config-mode
```

For example, to specify that a peer 23.34.45.56 uses the pre-shared key *mysecretkey* for IKE authentication and needs to be excluded from IKE mode configuration, we can use the following command:

```
isakmp key mysecretkey address 23.34.45.56 255.255.255.255 no-config-mode
```

Extended Authentication

IKE Extended Authentication (*xauth*) is an enhancement to IKE and is currently a draft RFC. *Xauth* is useful when configuring the Cisco software VPN client to

access the PIX firewall because it allows authentication to be performed after IKE Phase 1 and before Phase 2. Without *xauth*, IKE can only authenticate a device, not a user. With *xauth*, IKE is enhanced to support user authentication as well by allowing the server to request a username and password from the client. On the PIX firewall, the user is verified against an external RADIUS or TACACS+ server. (Local authentication cannot be used.) If verification fails, the IKE SA for this connection is deleted and the IPsec SAs will not be established. *Xauth* negotiation is performed before IKE mode configuration.

Before you enable *xauth*, you must define an AAA server group with AAA servers using the following commands:

```
aaa-server <group_tag> protocol <auth_protocol>
aaa-server <group_tag> [(interface)] host <server_ip> [<key>] [timeout
    <seconds>]
```

For example:

```
PIX1(config)# aaa-server vpnauthgroup protocol radius
PIX1(config)# aaa-server vpnauthgroup (inside) host 192.168.2.33 secretkey
    timeout 60
```

This command specifies that the RADIUS server 192.168.2.33 is in the group *vpnauthgroup*, has key *secretkey,* and has a timeout of 60 seconds.

Xauth negotiation is enabled in the crypto map. This is done using the following command:

```
crypto map <map-name> client authentication <group_tag>
```

Map-name is the name of crypto map for which *xauth* is enabled; *group_tag* is the name of a previously defined AAA group. For example, the following command forces IKE negotiations under map *mymap* to use *xauth* and authentication will be performed using the previously defined server 192.168.2.33:

```
PIX1(config)# crypto map mymap client authentication vpnauthgroup
```

Xauth faces the same problems as IKE mode configuration when the same interface is used for termination of both clients with dynamic addresses and site-to-site tunnels. It is possible to use the same technique to exclude some IP addresses from *xauth* negotiation. The command for configured exceptions is:

```
isakmp key <keystring> address <ip-address> [<netmask>] no-xauth
```

For example:

```
PIX1(config)# isakmp key mysecretkey address 23.34.45.56 255.255.255.255
    no-xauth
```

VPN Groups

The last feature used in configuring VPN client support is VPN groups. A Cisco VPN client is supposed to log into one of these groups in order to download its security parameters from a VPN concentrator or, in our case, a PIX firewall. A group is configured on PIX using the *vpngroup* set of commands. There are several commands in this set:

```
vpngroup <group_name> address-pool <pool_name>
vpngroup <group_name> default-domain <domain_name>
vpngroup <group_name> dns-server <dns_ip_prim> [<dns_ip_sec>]
vpngroup <group_name> idle-time <idle_seconds>
vpngroup <group_name> max-time <max_seconds>
vpngroup <group_name> password <preshared_key>
vpngroup <group_name> pfs
vpngroup <group_name> split-tunnel <acl_name>
vpngroup <group_name> wins-server <wins_ip_prim> [<wins_ip_sec>]
```

Most of these commands are self-explanatory. The *default-domain* command sets a domain name to be assigned to an authenticated client; *dns-server* and *wins-server* are the default DNS server and WINS server to be used by the client; and *pfs* forces the use of Perfect Forward Secrecy by all clients authenticated against this group. The *idle-time* parameter sets maximum inactivity timeout, after which the client is disconnected. The default idle timeout is 1800 seconds. *Max-time* specifies maximum connection time, after which the client is forced to disconnect. Default connection time is unlimited.

The *password* command specifies an IKE pre-shared key. In reality, when a VPN client connects to the PIX, it specifies its group name and the PIX tries to perform IKE negotiation using this password as a shared IKE key. The group name and password can be set in VPN Dialer when creating an entry. (See the following section for VPN client configuration examples.) There is another option for assigning passwords (shared keys) for IKE authentication. It is possible to use a single pre-shared key for all possible peers using the following command:

```
isakmp key <keystring> address 0.0.0.0 netmask 0.0.0.0
```

This is called a *wildcard* IKE key, and it means that this key is used regardless of the peer's IP address. So if you do not want to set different keys for different

groups and just want to use *xauth*, for example, it will be easier to set one wildcard IKE shared key and not specify passwords in VPN group configuration.

If an IKE Phase 1 negotiation is successful, IKE mode configuration is performed (it always has to be configured when a VPN client is used) and possibly *xauth* too (if configured, it is an optional feature). During IKE mode configuration, a client is assigned an internal IP address using either one common pool (this pool is defined as described previously in the "Mode Configuration" section and it must be in *initiate* mode) or the pool specific for this VPN group. Group-specific pools are defined by the following command:

```
vpngroup <group_name> address-pool <pool_name>
```

NOTE

If a group with the name *default* is configured, it will match any group name suggested by the VPN client. When IKE negotiation starts, the PIX looks for the group with the name suggested by client first, then for this *default* group. If neither is found, negotiation fails.

After IKE negotiation using pre-shared keys and optional extended authentication (*xauth*) succeeds, PIX downloads all parameters defined for this VPN group to the client and an IPsec tunnel is established. By default, all traffic from the computer where the VPN client is installed is tunneled to the PIX. One of the problems arising due to this process is that the client's Internet access will be terminated, because all client traffic is sent to the PIX. It is possible, however, to separate the traffic into two parts: one that should be tunneled and one that will be transmitted in the clear. This is done with the command:

```
vpngroup <group_name> split-tunnel <acl_name>
```

This command specifies an access list, which defines the traffic to be tunneled. It is a usual access list with the different meaning of a *permit* statement. *Permit* lines mean that matched traffic should be tunneled from client to PIX. If the destination matches a *deny* statement or does not match anything in this access list at all, the IP packet will be transmitted by the client in the clear. Figure 7.18 shows a minimal configuration (with only pre-shared keys authentication, no *xauth*) of a VPN group on PIX1 with split tunneling and corresponding IPsec settings.

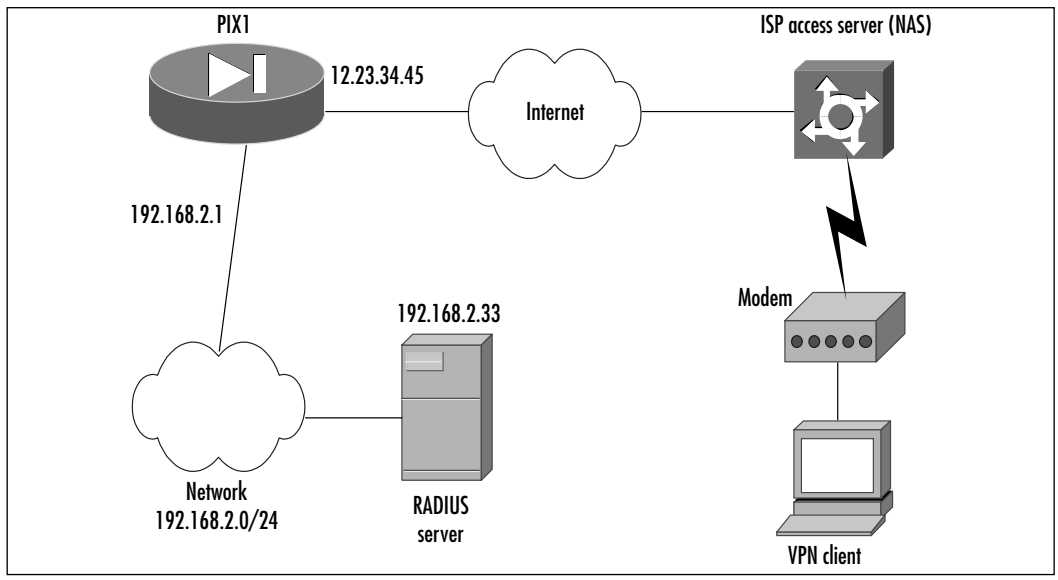
Figure 7.18 VPN Group Configuration

```
: only traffic between 192.168.2.0/24 and 192.168.10.0/24 will be tunneled
access-list 90 permit ip 192.168.2.0 255.255.255.0 192.168.10.0 255.255
     .255.0
: clients are assigned with ip addresses from 192.168.10.0/24
ip local pool vpnpool 192.168.10.1-192.168.10.254
: common dynamic map settings
crypto ipsec transform-set vpnset esp-des esp-sha-hmac
crypto dynamic-map dynmap 20 set transform-set vpnset
crypto map dialinmap 10 ipsec-isakmp dynamic dynmap
: pix will initiate ike mode configuration
crypto map dialinmap client configuration address initiate
crypto map dialinmap interface outside
: usual isakmp settings
isakmp enable outside
isakmp identity hostname
isakmp policy 5 authentication pre-share
isakmp policy 5 encryption 3des
isakmp policy 5 hash md5
isakmp policy 5 group 1
: vpn group "mygroup" is defined
: clients will be assigned ip addresses from "vpnpool"
vpngroup mygroup address-pool vpnpool
: dns, wins and domain are pushed to the client
vpngroup mygroup dns-server 192.168.2.33
vpngroup mygroup wins-server 192.168.2.34
vpngroup mygroup default-domain securecorp.com
: splitting according to access list 90 is defined
vpngroup mygroup split-tunnel 90
: timeouts are defined
vpngroup mygroup idle-time 1800
vpngroup mygroup max-time 86400
: ike shared key for this group is defined.
: it is actually shown as ****** in the PIX configuration
vpngroup mygroup password mypassword
```

Sample Configurations of PIX and VPN Clients

In this section, we consider a full configuration example of the PIX and a VPN client. Our example uses IKE with pre-shared keys, IKE mode configuration, and extended authentication (*xauth*) of the client against an internal RADIUS server. After that, we briefly discuss the changes needed in order to use digital certificates for IKE authentication. The network setup is shown in Figure 7.19.

Figure 7.19 Network Setup for Cisco VPN Client Configuration



Clients will be assigned IP addresses from the pool 192.168.10.1–192.168.10.254, and IKE authentication will use a wildcard key. Only the default VPN group will be configured. Configuration (assuming that PIX IP addresses are already configured) starts with defining an authentication server:

```
PIX1(config)# aaa-server vpnauthgroup protocol radius
PIX1(config)# aaa-server vpnauthgroup (inside) host 192.168.2.33 abcdef
    timeout 5
```

Next an IKE policy is configured (3DES encryption and MD5 hashing):

```
PIX1(config)# isakmp enable outside
PIX1(config)# isakmp policy 10 encryption 3des
PIX1(config)# isakmp policy 10 hash md5
PIX1(config)# isakmp policy 10 authentication pre-share
```

Cisco VPN client 3.*x* requires use of Diffie-Hellman Group 2 (1024-bit keys), not the default Group 1 (768-bit keys):

```
PIX1(config)# isakmp policy 10 group 2
```

A wildcard pre-shared key is configured, so all clients will use the same key:

```
PIX1(config)# isakmp key mysecretkey address 0.0.0.0 netmask 0.0.0.0
```

An access list for split tunneling is configured. Only traffic to or from network 192.168.2.0/24 will be protected:

```
PIX1(config)# access-list 80 permit ip 192.168.2.0 255.255.255.0 192.168
    .10.0 255.255.255.0
```

No-NAT is configured for IPsec traffic:

```
PIX1(config)# nat (inside) 0 access-list 80
```

Transform sets and crypto maps are configured and applied. This is a simple crypto map with only a dynamic map as a subentry.

```
PIX1(config)# crypto ipsec transform-set strong esp-3des esp-sha-hmac
PIX1(config)# crypto dynamic-map cisco 10 set transform-set strong
PIX1(config)# crypto map partner-map 20 ipsec-isakmp dynamic cisco
PIX1(config)# crypto map partner-map interface outside
```

Xauth is enabled for this map:

```
PIX1(config)# crypto map partner-map client authentication authserver
```

IKE mode configuration is enabled and an IP pool is created:

```
PIX1(config)# ip local pool dealer 192.168.10.1-192.168.10.254
PIX1(config)# isakmp client configuration address-pool local dealer
    outside
PIX1(config)# crypto crypto map partner-map client configuration address
    initiate
```

Initiate mode is optional for VPN client 3.*x* but must be used with clients version 2.*x*. The preceding two lines set global IKE mode configuration settings. They can be substituted by one command:

```
PIX1(config)# vpngroup default address-pool dealer
```

The difference is subtle here, because we configure the default group and its setting will be applied for any group name supplied by the VPN client. If you configure global IKE mode, it will also be applied to site-to-site tunnel

endpoints, so if you have any, you might need to exclude them. If there is none, there is no difference at all. A good way to have a simple configuration in case you have both site-to-site tunnels and VPN clients can be to use the default VPN group and define IKE mode configuration only for this group; it will not affect site-to-site gateways then.

Other VPN group settings are configured:

```
PIX1(config)# vpngroup default dns-server 192.168.2.44
PIX1(config)# vpngroup default wins-server 192.168.2.45
PIX1(config)# vpngroup default default-domain securecorp.com
PIX1(config)# vpngroup default split-tunnel 80
PIX1(config)# vpngroup default idle-time 1800
```

IPsec connections are implicitly permitted:

```
PIX1(config)# sysopt connection permit-ipsec
```

Figure 7.20 shows the full configuration of PIX1.

Figure 7.20 PIX1 Configuration

```
nameif ethernet0 outside security0
nameif ethernet1 inside security100
nameif ethernet2 dmz security10
enable password 8Ry2YjIRX7RXXU24 encrypted
passwd 2KFQnbNIdIXZJH.YOU encrypted
hostname PIX1
domain-name securecorp.com
fixup protocol ftp 21
fixup protocol http 80
fixup protocol smtp 25
fixup protocol h323 1720
fixup protocol rsh 514
fixup protocol sqlnet 1521
names
pager lines 24
no logging on
interface ethernet0 auto
interface ethernet1 auto
interface ethernet2 auto
mtu outside 1500
mtu inside 1500
```

Continued

Figure 7.20 Continued

```
mtu dmz 1500
ip address outside 12.23.34.54 255.255.255.0
ip address inside 192.168.2.1 255.255.255.0
no failover
failover ip address outside 0.0.0.0
failover ip address inside 0.0.0.0
arp timeout 14400
nat (inside) 1 0.0.0.0 0.0.0.0 0 0
access-list 80 permit ip 192.168.2.0 255.255.255.0 192.168.10.0 255.255
     .255.0
nat (inside) 0 access-list 80
global (outside) 1 12.23.34.55
route outside 0.0.0.0 0.0.0.0 12.23.34.254 1
timeout xlate 3:00:00 conn 1:00:00 half-closed 0:10:00 udp 0:02:00
timeout rpc 0:10:00 h323 0:05:00
timeout uauth 0:05:00 absolute
ip local pool dealer 192.168.10.1-192.168.10.254
aaa-server TACACS+ protocol tacacs+
aaa-server RADIUS protocol radius
aaa-server authserver protocol radius
aaa-server authserver (inside) host 192.168.2.33 abcdef timeout 5
no snmp-server location
no snmp-server contact
snmp-server community public
no snmp-server enable traps
crypto map partner-map client configuration address initiate
crypto ipsec transform-set strong esp-3des esp-sha-hmac
crypto dynamic-map cisco 10 set transform-set strong-des
crypto map partner-map 20 ipsec-isakmp dynamic cisco
crypto map partner-map client authentication authserver
crypto map partner-map interface outside
isakmp key mysecretkey address 0.0.0.0 netmask 0.0.0.0
isakmp enable outside
isakmp policy 10 authentication pre-share
isakmp policy 10 encryption 3des
isakmp policy 10 hash md5
isakmp policy 10 group 2
vpngroup default address-pool dealer
```

Continued

Figure 7.20 Continued

```
vpngroup default dns-server 192.168.2.44
vpngroup default wins-server 192.168.2.45
vpngroup default default-domain securecorp.com
vpngroup default split-tunnel 80
vpngroup default idle-time 1800
sysopt connection permit-ipsec
telnet timeout 5
terminal width 80
```

The Cisco VPN client is configured as follows. Select **Start | Programs | Cisco Systems VPN Client | VPN Dialer** and select **New** to create a new connection entry (see Figure 7.21).

Figure 7.21 Creating a New Connection Entry

The New Connection Entry wizard starts. After asking you to name this connection (enter whatever you want here), it asks for the IP address of the server. In our case, this is the IP address of the outside interface of PIX1, where the tunnel will be terminated (see Figure 7.22). Enter the IP address, and click **Next**.

Figure 7.22 Entering the Server IP Address

Next you need to enter the name of the group and the shared IKE secret. In our case, because we defined a *default* group on PIX, the group name does not really matter; any name will match the *default* group. If, on the other hand, we specified another name in the PIX configuration, we would have needed to specify exactly the same name on this screen. The password is the shared key *mysecretkey* (see Figure 7.23). Again, if were using a separate password for each VPN group, the password that corresponds to the group's name should be entered here.

After clicking **Next** and then **Finish**, we are done. It is possible to modify this entry's properties by clicking **Options | Properties** in the main window of VPN Dialer. Among other properties, it is possible to change group name and password, set timeouts, and select the dialup connection that must be dialed before establishing the tunnel.

Now you need to select the connection you just created and click the **Connect** button (see Figure 7.24).

If network connectivity is correct (nothing blocks IKE port UDP/500 between your host and the firewall, for example), IKE negotiation starts. It checks for a shared secret first, then *xauth* starts and the VPN client displays a new window asking you to enter a username and a password. After you do this, the username and password are checked against the RADIUS server specified in the PIX configuration. If everything is correct, the tunnel is established and the PIX downloads settings such as an internal IP address, DNS, and WINS settings to the VPN client.

Figure 7.23 Specifying the VPN Group and the IKE Shared Secret

Figure 7.24 Connecting to the Server

You can check that the connection works by pinging some internal PIX hosts from the client computer. It is also possible to monitor established tunnels by the usual PIX *debug* commands such as *debug vpdn event*, *debug vpdn error*, and *debug vpdn packet*. You can also use all IPsec and IKE-related *debug* commands.

In order to use digital certificates, the CA is configured (we will use VeriSign as before) and IKE is reconfigured correspondingly. The whole configuration changes just a few commands. See Figure 7.25 for a listing of PIX configurations with new or changed commands in italics.

Figure 7.25 PIX1 Configuration for Use with IKE CA Authentication

```
nameif ethernet0 outside security0
nameif ethernet1 inside security100
nameif ethernet2 dmz security10
enable password 8Ry2YjIRX7RXXU24 encrypted
passwd 2KFQnbNIdIXZJH.YOU encrypted
hostname PIX1
domain-name securecorp.com
fixup protocol ftp 21
fixup protocol http 80
fixup protocol smtp 25
fixup protocol h323 1720
fixup protocol rsh 514
fixup protocol sqlnet 1521
names
pager lines 24
no logging on
interface ethernet0 auto
interface ethernet1 auto
interface ethernet2 auto
mtu outside 1500
mtu inside 1500
mtu dmz 1500
ip address outside 12.23.34.54 255.255.255.0
ip address inside 192.168.2.1 255.255.255.0
no failover
failover ip address outside 0.0.0.0
failover ip address inside 0.0.0.0
arp timeout 14400
```

Continued

Figure 7.25 Continued

```
nat (inside) 1 0.0.0.0 0.0.0.0 0 0
access-list 80 permit ip 192.168.2.0 255.255.255.0 192.168.10.0 255.255
    .255.0
nat (inside) 0 access-list 80
global (outside) 1 12.23.34.55
route outside 0.0.0.0 0.0.0.0 12.23.34.254 1
timeout xlate 3:00:00 conn 1:00:00 half-closed 0:10:00 udp 0:02:00
timeout rpc 0:10:00 h323 0:05:00
timeout uauth 0:05:00 absolute
ip local pool dealer 192.168.10.1-192.168.10.254
aaa-server TACACS+ protocol tacacs+
aaa-server RADIUS protocol radius
aaa-server authserver protocol radius
aaa-server authserver (inside) host 192.168.2.33 abcdef timeout 5
no snmp-server location
no snmp-server contact
snmp-server community public
no snmp-server enable traps
crypto map partner-map client configuration address initiate
crypto ipsec transform-set strong esp-3des esp-sha-hmac
crypto dynamic-map cisco 10 set transform-set strong-des
crypto map partner-map 20 ipsec-isakmp dynamic cisco
crypto map partner-map client authentication authserver
crypto map partner-map interface outside
isakmp enable outside
isakmp policy 10 authentication rsa-sig
isakmp policy 10 encryption 3des
isakmp policy 10 hash md5
isakmp policy 10 group 2
vpngroup mygroup address-pool dealer
vpngroup mygroup dns-server 192.168.2.44
vpngroup mygroup wins-server 192.168.2.45
vpngroup mygroup default-domain securecorp.com
vpngroup mygroup split-tunnel 80
vpngroup mygroup idle-time 1800
ca identity verisign 205.139.94.230
ca configure verisign ca 1 20 crloptional
sysopt connection permit-ipsec
```

Continued

Figure 7.25 Continued

```
telnet timeout 5
terminal width 80
```

The group name was changed from the default because in digital certificates the name of the group must match the Organizational Unit section of the Cisco VPN client certificate. This certificate must be obtained and installed before configuring the connection entry. The process of obtaining the certificate is described in VPN client documentation at www.cisco.com/univercd/cc/td/doc/product/vpn/index.htm. Client certificates are managed by Certificate Manager, which is installed together with the VPN client.

Client configuration after the certificate has been obtained does not change much compared to the case of pre-shared keys. Only the step shown in Figure 7.23 changes; you need to select your certificate instead of a name for the group. See Figure 7.26.

Figure 7.26 Using a Digital Certificate for IKE Authentication

Connectivity can be verified as before, and troubleshooting uses the same PIX *debug* commands.

Summary

Virtual private networks are used to securely tunnel traffic between two sites over a public network such as the Internet. VPNs are commonly used to connect branch offices, mobile users, and business partners. The two common types of VPNs are site-to-site and remote access. The PIX firewall supports both types of VPN using various protocols: IPsec, L2TP, and PPTP.

The most robust tunneling solution for IP networks is the IPsec suite of protocols. It was developed by IETF as part of IPv6. IPsec operates at Layer 3 of the OSI model, which means that it can protect communications from the network layer (IP) and up. IPsec specifies encryption and authentication algorithms, AH and ESP protocols used for tunneling itself, and the IKE/ISAKMP key management protocol. IPsec's main goals are data confidentiality, data integrity, data origin authentication, and antireplay service.

When a site-to-site IPsec tunnel is configured on a PIX firewall, one of two main methods of IKE authentication are used: pre-shared keys or digital certificates. The former is simpler to set up, but it lacks scalability offered by the digital certificate solution. It is also possible to not use IKE at all. In this configuration, all IPsec parameters can be configured manually; this is called *manual IPsec*. There are two encapsulation modes in IPsec: tunnel and transport. The PIX almost always uses tunnel mode, with the exception of L2TP tunneling, where transport mode is used.

In the second type of VPN, remote clients connect to a gateway. The PIX supports various protocols for this type of VPN. Point-to Point Tunneling Protocol (PPTP) uses PPP encapsulation for tunneling traffic from the client to PIX and can transport any Layer 3 protocol supported by the PPP specification. PPTP is a Layer 2 tunneling protocol in terms of ISO/OSI model, whereas IPsec works with Layer 3 tunnels.

Another type of Layer 2 tunneling is Layer 2 Tunneling Protocol (L2TP). The PIX uses it together with IPsec in transport mode in order to encrypt and authenticate packets. L2TP configuration resembles a combination of the configurations of IPsec and PPTP. Both PPTP and L2TP protocols are supported by the internal Windows 2000 VPN client.

Cisco has its own software VPN client that provides full IPsec features when working with the PIX firewall. It can perform IKE authentication with both pre-shared keys and digital certificates. The PIX uses two extensions to IKE to provide VPN clients with an internal IP address (IKE mode configuration) and perform extra authentication of clients during IKE negotiation using Extended Authentication (*xauth*).

Solutions Fast Track

IPsec Concepts

- ☑ The main features of IPsec are data confidentiality, data integrity, data origin authentication, and antireplay service.
- ☑ IPsec specifies low-level encryption and authentication algorithms, IP encapsulation protocols, and key management tools.
- ☑ There are two types of VPN: site-to-site and remote access.
- ☑ IPsec can be used in two modes: transport and tunnel. All PIX site-to-site VPNs use tunnel mode.

Configuring Site-to-Site IPsec Using IKE

- ☑ Site-to-site tunnels can use IKE in pre-shared keys mode or digital certificates. The former is simpler to configure, but the latter provides more scalability.
- ☑ The PIX has separate configurations for IKE parameters and for the rest of IPsec, such as the set of encryption protocols and security policies for traffic protection.
- ☑ It is possible to implicitly allow all authenticated IPsec traffic through a PIX firewall, thus not requiring any special conduits for each tunnel. This is accomplished using the *sysopt connection permit-ipsec* command.

Configuring Point-to-Point Tunneling Protocol

- ☑ PPTP is an encapsulation of traffic using PPP and then Generic Routing Encapsulation (GRE). Since it operates at Layer 2, it can also tunnel protocols other than IP.
- ☑ PPTP is generally used for remote access networks and is supported by the Windows 2000 internal VPN client.
- ☑ Authentication for PPTP connections is provided on the PIX and can be performed against the local database or an external AAA server.

Configuring Layer 2 Tunneling Protocol with IPsec

- ☑ Layer 2 Tunneling Protocol (L2TP) is another Layer 2 tunneling protocol that can tunnel non-IP protocols. Using L2TP is the only time when the PIX can be configured in IPsec transport mode.
- ☑ Windows 2000 internal client supports only digital certificates authentication, although Microsoft provides some documentation on possible ways to support pre-shared keys IKE authentication. L2TP users are further authenticated by PPP means such as PAP, CHAP, or MSCHAP.
- ☑ Encryption, packet authentication, and antireplay services are provided by an IPsec tunnel.

Configuring Site-to-Site IPsec Without IKE (Manual IPsec)

- ☑ It is possible to configure IPsec without IKE. This is also known as *manual IPsec*.
- ☑ Manual IPsec is difficult to scale and is not recommended. It is also less secure because there is no SA lifetime and PFS cannot be enabled.
- ☑ For manual IPsec to function, an inbound session key and an outbound session key must be configured manually.

Configuring Support for the Cisco Software VPN Clients

- ☑ Cisco VPN client 3.*x* supports all IPsec features, including IKE with pre-shared keys or digital certificates.
- ☑ The Cisco PIX firewall uses extensions to IKE mode configuration and Extended Authentication to assign remote clients internal IP addresses, download configuration settings to them, and perform additional authentication.
- ☑ User authentication using *xauth* can only be performed by external AAA servers. The local PIX database cannot be used.

- ☑ The Cisco VPN client, when installed, takes over the internal Windows 2000 IPsec client so that the latter cannot function correctly.
- ☑ It is possible to specify which traffic has to be tunneled through the IPsec connection and which must be transmitted in clear so that user Internet and LAN connection does not cease after the tunnel is established.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: Which IKE lifetime parameters are supported?

A: Although there are two parameters, *time lifetime* and *volume lifetime*, only the former is currently supported, so the output of *show isakmp policy* will always show a *no volume limit* setting.

Q: I am having trouble connecting a Windows 200 VPN client to a PIX L2TP gateway. What can be wrong?

A: Such problems are commonly caused by a mismatch in either the transform sets or the IKE or IPsec SA lifetimes. They should be negotiated in theory, but it is better to configure them to match exactly.

Q: All IPsec connections are dropped when I reapply a crypto map to the interface. Is this normal behavior?

A: Yes. When a crypto map is applied to an interface, all internal IPsec-related structures such as SPD and SAD are reinitialized, so all SAs are deleted and all tunnels are dropped. Unfortunately, for any change in a crypto map to become effective, it has to be reapplied.

Q: My Internet connectivity drops after I establish a VPN connection with PIX using a VPN client. What is the cause of this problem?

A: Most probably you did not specify split tunneling in PIX configuration, so all your traffic is directed to PIX and therefore you cannot reach the Internet. Configure split tunneling in order to tunnel only the interesting traffic and let everything else be transmitted in the clear.

Q: What are the specifics for configuring the PIX to support VPN client 2.*x* and 3.*x*?

A: VPN client v3.*x* requires the use of Diffie-Hellman Group 2 in IKE exchange. VPN client version 2.*x* requires that IKE mode configuration be initiated by the PIX because it cannot initiate this process by itself.

Chapter 8

Configuring Failover

Solutions in this chapter:

- Failover Concepts
- Standard Failover Using a Failover Cable
- LAN-Based Failover

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

Many enterprises have high requirements for availability. In many environments, providing 99.99 percent uptime is part of the service-level agreement (SLA), which equals less than 53 minutes of downtime a year. In order for this uptime requirement to be met, high availability through redundancy and failover must be implemented. To support high availability, the PIX firewall provides both stateless and stateful failover capabilities.

In this chapter, you will learn how failover works on the PIX firewall. We will go through various configuration examples to learn all types of failover. You will also learn about LAN-based failover operation, which is one of the newer features available on the PIX firewall.

Failover Concepts

The failover feature of the Cisco PIX firewall provides the ability to deal with firewall failures. This is accomplished by running a second PIX firewall that automatically takes over in case the active firewall fails. Failover works with two, and exactly two, firewalls. When one of these firewalls fails, the other one takes over the failed firewall's functions. Failover works with all interface types. The two firewalls must be identical in the following ways:

- Same model of firewall (for example, a PIX 515 cannot be used with a PIX 515E)
- Same amount of flash memory and RAM
- Same software version (for example, software version 6.1 cannot be used with software version 6.2)
- Same number and types of interfaces
- Same activation key type (for example, DES or 3DES support)

In addition, there are some licensing restrictions for using failover:

- The primary firewall must be running an *unrestricted* license.
- The secondary firewall must be running either an *unrestricted* or a *failover-only* license.

Failover is only supported on the high-end models of the PIX firewall, such as the PIX 515, 515E, 520, 525, and 535. It is *not* supported on the PIX 501, 506, and 506E.

Firewalls with failover-only licenses can usually be obtained from Cisco at very low prices. Cisco also offers bundled firewall pricing, selling two PIX firewall units (one with an unrestricted license and the other with a failover-only license) for low prices.

Designing & Planning…

Load Balancing vs. Redundancy

The failover feature in PIX firewalls only provides support for redundancy. One unit acts as the active firewall, and the other one runs in standby mode. It is not possible to run both firewalls in active mode at the same time. If you want to increase capacity by using two or more firewalls in active mode, you should consider purchasing some load-balancing equipment. Load balancers, such as the Cisco Content Services Switch (CSS) 11000 series, provide the ability to load-balance network traffic to multiple PIX firewalls in order to provide increased capacity and higher combined throughput rates. Be careful to configure your load balancers to work on a *per-conversation* basis. If they are configured to work on a *per-packet* basis, the stateful inspection feature of the PIX firewall will end up denying valid traffic.

NOTE

A firewall with a failover-only license is meant to be used as a secondary firewall for failover only, not for standalone operation. If used in standalone mode, the firewall will reboot once every 24 hours and display the following message on the console:

```
=========================NOTICE ==========================
    This machine is running in secondary mode without
      a connection to an active primary PIX. Please
      check your connection to the primary system.
                    REBOOTING....
============================================================
```

The reboots will continue until the firewall is re-configured for operation as a failover unit.

When you configure failover, one firewall is designated as *primary*, and the other one is designated as *secondary*. In normal mode of operation, when everything is functioning properly, the primary firewall is *active* and handles all the network traffic. The secondary firewall sits in *standby* state and is ready to take over the functions of the primary firewall in case the primary fails. When the primary fails, the secondary firewall becomes active, and the primary goes into the standby state. A standby firewall can also fail. If the standby firewall fails, the primary remembers this and the secondary is disallowed to ever take control, so failover will not occur even if the primary firewall later fails. Although the firewalls may switch the active and standby roles, the primary and secondary never change. This terminology of *primary, secondary, active,* and *standby* is extremely important to understand as they relate to other failover concepts.

This brings us to a very important question: When is a firewall considered failed? Failure happens when any of the following conditions occurs:

- Block memory is exhausted for 15 consecutive seconds or longer on the active PIX firewall.
- The link status of any of the network interfaces on the active PIX goes down for more than twice the poll interval. This does not include interfaces that are administratively down.
- Hello packets are constantly exchanged between the primary and secondary PIX firewalls over all network interfaces. (They are sent out every 15 seconds by default, but this interval can be tuned.) If no hello messages are received for two poll intervals, the interface that did not respond is put into testing mode. If the interface does not pass testing, it as well as the firewall are considered failed.
- Hello packets are also exchanged between the primary and secondary PIX firewalls over the failover serial cable. If the standby firewall does not hear from the active firewall for two poll intervals and the failover cable status is declared okay, the standby PIX firewall considers the active PIX failed and becomes active itself. Furthermore, if the active unit does not hear from the standby firewall for two poll intervals, it considers the standby unit as failed.
- If the standby firewall detects that the active firewall has been powered off or rebooted, the standby becomes active. If the failover cable is unplugged, no failover occurs.

NOTE

The failover cable is designed to be intelligent enough to distinguish between a power failure on the other unit, a cable unplugged from this unit, or a cable unplugged from the other unit. Therefore, if the cable is unplugged from either unit, no failover occurs, but a syslog message is generated. However, if the active firewall is powered off (either gracefully using a *reload* command or through a power failure), the standby unit assumes the active state.

There are two types of failover—standard failover and LAN-based failover—and the two function in a similar manner. The primary difference between them is the means used to exchange failover information between the primary and secondary firewalls. In standard failover, a special serial cable is used to connect the two firewalls. This cable is known as the *failover cable*. The failover cable is a Cisco proprietary modified RS-232 cable that is used specifically for PIX firewalls. In LAN-based failover, instead of the failover serial cable, a dedicated Ethernet link is used to exchange failover information.

The failover information exchange over the serial failover cable (or the failover Ethernet link in LAN-based failover) includes:

- The MAC addresses of the firewalls
- Hello (keepalive) packets
- State information (active or standby)
- Network interface link status
- Configuration replication

Communication over the failover cable is performed using messages, and each message must be acknowledged. If a message is not acknowledged by the other firewall within 3 seconds, it is retransmitted. After five retransmissions without an acknowledgment, the firewall that is not acknowledging messages is declared failed.

Configuration Replication

Configuration replication is the process by which the configuration from the primary PIX firewall is replicated to the secondary firewall. When the replication process begins, a "Sync started" message is displayed on the console, and similarly, when replication completes, a "Sync complete" message is displayed on the

console. The replication process occurs from memory to memory (running-config to running-config) only and is not saved in flash. Therefore, after replication is complete, a *write memory* command should be issued on both the active and standby firewalls.

The replication process is automatically performed when:

- The standby PIX completes initial bootup. The primary firewall replicates its entire configuration to the secondary firewall.
- Commands are typed in on the active PIX firewall. As each command is entered on the active PIX firewall, it is sent to the standby across the failover connection.
- The *write standby* command is executed on the active PIX firewall. This forces the entire configuration to be replicated from the primary PIX firewall to the standby.

Any configuration changes made on the standby firewall are not replicated to the primary. If you try to enter commands on the standby firewall, the PIX will warn you that you are trying to configure the wrong firewall.

IP and MAC Addresses Used for Failover

For each network interface on which you want failover configured, you need to reserve two IP addresses. One IP address is for the primary firewall, and one IP address is for failover. When functioning properly, the primary firewall will use its system IP and MAC addresses, and the secondary firewall will use the failover IP and MAC addresses. When failover occurs, the primary firewall fails and the secondary firewall becomes active, and the IP and MAC addresses are swapped. In other words, the secondary firewall (now active) assumes the system IP and MAC addresses of the primary firewall. The primary firewall (now standby) assumes the failover IP and MAC addresses of the secondary firewall. Since the MAC addresses of the firewalls change in addition to the IP addresses, hosts connecting through the firewall do not have to re-ARP.

By default, the MAC addresses on the active firewall are the burned-in MAC addresses from the NICs of the primary firewall, and the MAC addresses on the standby firewall are the burned-in MAC addresses from the NICs on the secondary firewall. Instead of using these burned-in addresses, you have the option to use a virtual MAC address. If you decide to do this, for each interface you can assign virtual MAC addresses using the following command:

```
failover mac address <if_name> <active_mac> <standby_mac>
```

For example:

```
PIX1(config)# failover mac address inside 00c0.1715.3341 00c0.1715.3342
```

To remove a virtual MAC address, use the *no* form of the command.

Failure Detection

The primary and secondary firewalls exchange hello packets with each other over the failover cable as well as all network interfaces. These hellos are exchanged every 15 seconds by default. To change the hello interval, use the following command:

```
failover poll <seconds>
```

The minimum value for *seconds* is 3 seconds, and the maximum is 15 seconds. With a lower hello interval, failure will be detected faster, but there is also the danger of unnecessary failover occurring when the network is experiencing temporary congestion.

The failover feature on the PIX firewall monitors failover communication hello packets as well as the power status on the other firewall. If a failure is detected and it is not because of a power loss or reboot of the secondary firewall, the PIX firewall (primary or secondary, whichever detects a failure) performs a series of tests to determine which firewall has failed. The tests begin when hello messages are not heard for two consecutive poll intervals, which is set to 15 seconds by default. The idea behind each test is to look for network traffic. For each of these tests, if one firewall receives network traffic during a test and the other firewall does not, the firewall that has not received any traffic is considered failed. If neither firewall receives any traffic, the next test in the series is performed. The following four tests are used:

- **Link up/down** The firewall tests the network link state to ensure it is up. This test finds issues such as a cable being unplugged, a hub/switch port going bad, or a hub/switch failure. If the interface passes this test, the PIX starts the network activity test. Otherwise, the interface and the corresponding firewall are considered failed.
- **Network activity** The firewall listens for network activity for up to 5 seconds. If any packets are received during this testing, the interface is considered operational and testing stops. If no activity is found, the PIX firewall starts the ARP test.
- **ARP** If the network activity test fails, the Address Resolution Protocol (ARP) test is performed. The PIX takes the 10 most recent entries

added to its ARP table and sends ARP requests for each one in order to stimulate some network traffic. After sending each request, the PIX monitors all received traffic for up to 5 seconds. If no traffic is received, the PIX moves on to the next entry in the list. If at any time during the test network traffic is received, the interface is considered operational and testing stops. If the list is exhausted and no traffic has been received, the PIX starts the broadcast ping test.

- **Broadcast ping** The firewall sends out a broadcast ping on the interface and looks at all packets received for up to 5 seconds after the ping was sent. If any packets are received, the PIX firewall declares the interface operational and stops the test. If, however, no packets are received, the firewall starts testing all over again with the ARP test.

NOTE

All corresponding interfaces (which are not administratively shut down) on both firewalls must be able to communicate with each other, even if they're not used. For example, they can be connected with crossover cables or plugged into the same switch. Otherwise, the tests will fail.

Stateful Failover

As of software version 5.1, the PIX firewall supports stateful failover. Before the stateful failover feature, when the primary firewall failed and the secondary became active, all active connections through the firewall were dropped, and applications needed to start new connections through the firewall. If configured, the stateful failover feature can eliminate this problem. With stateful failover enabled, the primary firewall constantly replicates its TCP connection table to the secondary PIX firewall. If the primary firewall fails, the secondary firewall already has the connection table and therefore no connections are lost. Client applications continue to function without interruption, unaware that a failover situation occurred.

When using stateful failover, in addition to the configuration, the following information is replicated to the standby PIX firewall:

- The translation (*xlate*) table with static and dynamic translations
- The TCP connection table (including timeout information for each connection)

- The system clock and uptime information

Most UDP connections are not replicated, with the exception of certain multichannel protocols such as H.323. The following information is *not* replicated to the standby PIX firewall:

- ISAKMP and IPsec state information; this means that any ISAKMP and IPsec SAs are lost when failover occurs
- DHCP leases
- The user authentication (*uauth*) table; when failover occurs, any authenticated users must reauthenticate
- The routing table; this means that all dynamically learned routes (through RIP) must be relearned.
- The ARP table

By default, HTTP session information is not replicated. In PIX 6.2 and later, this feature can be enabled using the following command:

```
PIX1(config)# failover replicate http
```

You can verify the configuration of HTTP replication using the *show failover* command. To disable HTTP replication, use the *no* form of the command:

```
PIX1(config)# no failover replicate http
```

For stateful failover to work, a Fast Ethernet or Gigabit Ethernet interface on each firewall (primary and secondary) must be dedicated for the exclusive use of passing state information. (We refer to this as the *stateful failover interface.*) This interface must provide connectivity between the primary and secondary firewalls through one of the following methods:

- A crossover Ethernet cable
- A dedicated hub or switch, with no other hosts
- A dedicated VLAN on a switch with only the two ports connecting to the firewalls active in the VLAN

NOTE

It is recommended that the stateful failover interface be *at least* as fast as the fastest used interface on the firewall.

NOTE

Token Ring and FDDI interfaces are not supported for use as the dedicated stateful failover interface.

Standard Failover Using a Failover Cable

In standard failover, a special serial cable known as the *failover cable* is used to connect the primary and secondary PIX firewalls. One end of the failover cable is labeled *primary*, and the other end is labeled *secondary*. As common sense would dictate, the primary end of the cable should be connected to the firewall that you want to designate as primary, and the secondary end of the cable should be connected to the firewall that you want to designate as secondary. This cable should be connected when the secondary firewall is turned off.

The failover cable exchanges state data between the firewalls at 115Kbps. Communication over the failover cable includes:

- Hellos (keepalives)
- MAC address exchanges
- State (active vs. standby)
- Network link status
- Configuration replication

NOTE

Before software version 5.2 of the PIX firewall, the failover cable operated at 9600bps.

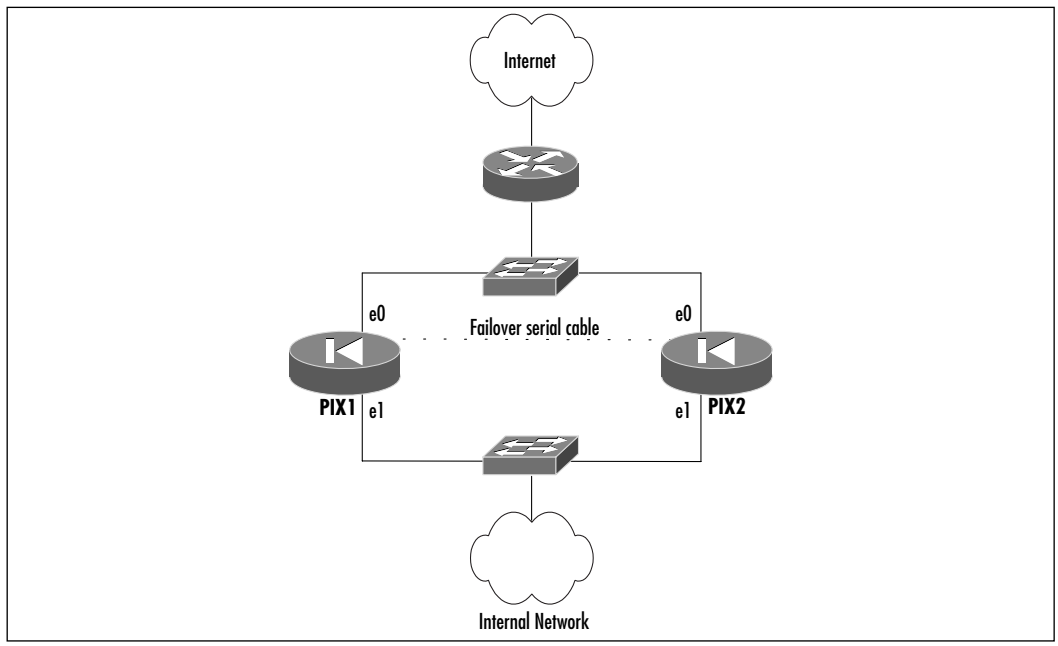
NOTE

Do *not* connect the failover cable backward. Doing so will cause a replication to occur from the secondary firewall to the primary firewall and erase your entire configuration!

Configuring and Enabling Failover

Failover configuration is straightforward and does not require many commands. In this section, we review a case study, configuring standard failover step by step. At each step of the configuration, we use *show* commands to check the status. The network topology is shown in Figure 8.1. In this example, PIX1 is the primary firewall, and PIX2 is the secondary firewall. There are two interfaces in use, ethernet0 (outside) and ethernet1 (inside).

Figure 8.1 Standard Failover Example



Before we start, we plug in the failover cable, being careful to connect the *primary* end into the primary firewall and the *secondary* end into the secondary firewall. Each interface on the primary firewall also needs to be connected to the corresponding interface on the secondary firewall through either a switch or a crossover cable. In this example, we are using Layer 2 switches, and all the ports on each switch are on the same VLAN. We also make sure that all the switches are configured and powered on and that all Ethernet cables are plugged in correctly. We leave the secondary firewall powered off, and we turn on the primary firewall. Next, we configure the clock on the primary firewall using the *clock* command.

NOTE

Do not power on the secondary firewall until the primary firewall is fully configured.

Cisco recommends that when you use failover, no network interface should be set for autonegotiation. In other words, do not use the *auto* or *1000auto* keywords in your interface configuration commands. Each interface involved in failover should be hardcoded for speed and duplex settings using the *10baset, 100basetx, 100full*, *1000basesx*, or *1000sxfull* keywords. Make sure that these settings match the hub or switch to which the interface is connected. In our example, we are using all 100Mbps interfaces, so we will hardcode the interfaces to 100Mbps full-duplex operation:

```
PIX1(config)# interface ethernet0 100full
PIX1(config)# interface ethernet1 100full
```

Of course, we also configured our switches for 100Mbps full duplex. Before enabling failover, we must first assign IP addresses to each interface on the primary firewall:

```
PIX1(config)# ip address inside 192.168.1.1 255.255.255.0
PIX1(config)# ip address outside 10.5.1.1 255.255.255.0
```

To verify the IP addresses, use the *show ip address* command:

```
PIX1# show ip address
System IP addresses:
        ip address outside 10.5.1.1 255.255.255.0
        ip address inside 192.168.1.1 255.255.255.0
Current IP addresses:
        ip address outside 10.5.1.1 255.255.255.0
        ip address inside 192.168.1.1 255.255.255.0
```

At this point, the current IP addresses on the primary firewall should be the same as the system IP addresses. When failover occurs, the current IP addresses will change to the failover IP addresses. Before we dive into the configuration, let's use the *show failover* command to check the current failover status:

```
PIX1# show failover
Failover Off
```

```
Cable status: Other side powered off
Reconnect timeout 0:00:00
Poll frequency 15 seconds
```

As shown in the first line in the command output, failover is currently not enabled. The second line in the command output shows us that the other end of the failover cable is connected correctly and that the secondary firewall is powered off.

To enable failover, we use the *failover* command on the primary firewall:

```
PIX1(config)# failover
```

Now we can use the *show failover* command on the primary firewall to verify that failover is enabled and that it is acting as the active firewall (see Figure 8.2).

Figure 8.2 Output of the show failover Command After Enabling Failover

```
PIX1# show failover
Failover On
Cable status: Other side powered off
Reconnect timeout 0:00:00
Poll frequency 15 seconds
        This host: primary - Active
                  Active time: 60 (sec)
                  Interface outside (10.5.1.1): Normal (Waiting)
                  Interface inside (192.168.1.1): Normal (Waiting)
        Other host: secondary - Standby
                  Active time: 0 (sec)
                  Interface outside (0.0.0.0): Unknown (Waiting)
                  Interface inside (0.0.0.0): Unknown (Waiting)
```

As shown in the command output here, the fifth line reads, "This host: primary – Active," which means that you are on the primary firewall and it is active for failover. Next, we configure the failover IP addresses using the *failover ip address* command. This needs to be done for each interface. Normally, in an unfailed state, these IP addresses will be assigned to their corresponding interfaces of the standby unit. Make sure that failover IP addresses are in the same subnet as the active IP addresses:

```
PIX1(config)# failover ip address inside 192.168.1.2
PIX1(config)# failover ip address outside 10.5.1.2
```

We can use the *show failover* command on the primary firewall again to verify the status of the failover IP addresses (see Figure 8.3). As you can see from the output of the command, under "Other host," the secondary firewall now has IP addresses for each interface.

Figure 8.3 Output of the show failover Command After Configuring Failover IP Addresses

```
PIX1# show failover
Failover On
Cable status: Other side powered off
Reconnect timeout 0:00:00
Poll frequency 15 seconds
        This host: primary - Active
                  Active time: 300 (sec)
                  Interface state (172.16.1.1): Normal (Waiting)
                  Interface outside (10.5.1.1): Normal (Waiting)
        Other host: secondary - Standby
                  Active time: 0 (sec)
                  Interface state (172.16.1.2): Unknown (Waiting)
                  Interface outside (10.5.1.2): Unknown (Waiting)
```

At this point, failover configuration is complete. Yes, it was that simple! We now need to power on the secondary firewall. After the secondary firewall boots up, the primary will detect it and will start to synchronize the configurations. You will see the following message on the console:

```
Sync Started
```

Once the synchronization is complete, you will see:

```
Sync Completed
```

We can use the *show failover* command on the primary firewall to verify the status (see Figure 8.4).

Figure 8.4 Output of the show failover Command After Completing the Configuration

```
PIX1# show failover
Failover On
Cable status: Normal
Reconnect timeout 0:00:00
Poll frequency 15 seconds
        This host: primary - Active
                  Active time: 350 (sec)
                  Interface state (172.16.1.1): Normal
                  Interface outside (10.5.1.1): Normal
        Other host: secondary - Standby
                  Active time: 0 (sec)
                  Interface state (172.16.1.2): Normal
                  Interface outside (10.5.1.2): Normal
```

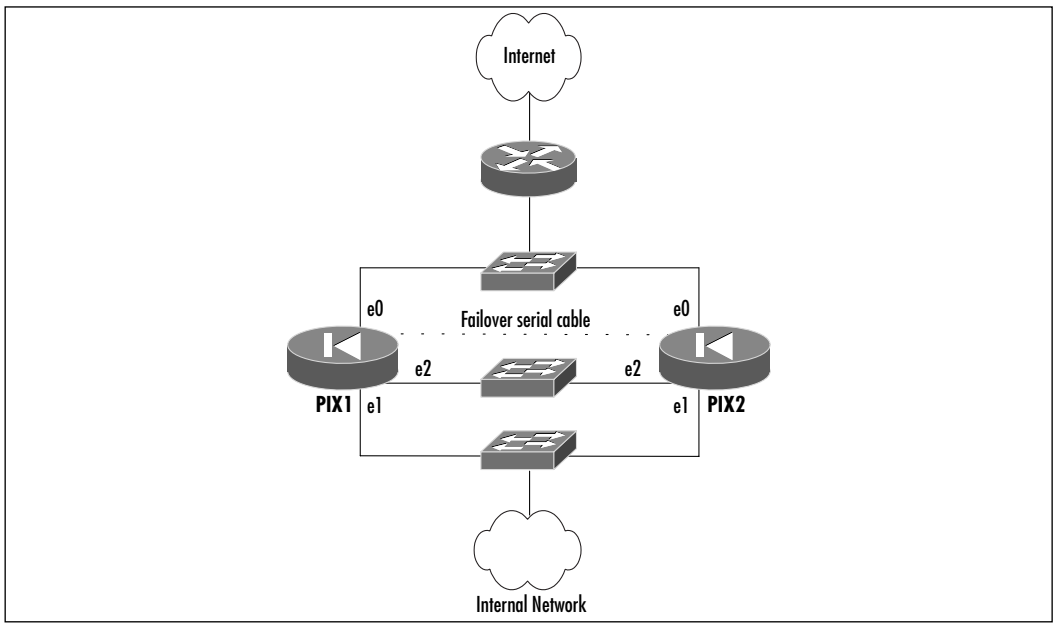
As shown in the command output, the *Unknown* status has changed to *Normal*. The cable status also displays as *Normal,* meaning that failover is operating normally. This is the output that you usually want to see on your primary firewall.

Now let's enable the stateful failover feature on these firewalls. First, we must set up a dedicated network link between the two firewalls that will be used for exchanging state information. As shown in Figure 8.5, we have selected ethernet2 on each firewall for this function and have connected a switch between the interface on both firewalls. (We could also have used a crossover cable instead of using a switch.)

We need to configure the interface settings for ethernet2, give it a name (we picked the name *state*), and assign system and failover IP addresses:

```
PIX1(config)# nameif ethernet2 state security25
PIX1(config)# interface ethernet2 100full
PIX1(config)# ip address state 172.16.1.1 255.255.255.0
PIX1(config)# failover ip address state 172.16.1.2
PIX2(config)# nameif ethernet2 state security25
PIX2(config)# interface ethernet2 100full
```

Figure 8.5 Standard Stateful Failover Example



After the interface is configured, there is only a single command to enter to make this the stateful failover interface:

```
PIX1(config)# failover link state
```

NOTE

The stateful failover interface (ethernet2 in our example) must have its MTU set to 1500 bytes or larger.

You can verify stateful failover operation using the *show failover* command (see Figure 8.6).

Figure 8.6 Output of the show failover Command After Enabling Stateful Failover

```
PIX1# show failover
Failover On
Cable status: Normal
```

Continued

Figure 8.6 Continued

```
Reconnect timeout 0:00:00
Poll frequency 3 seconds
        This host: Primary - Active
                Active time: 400 (sec)
                Interface state (172.16.1.1): Normal
                Interface outside (10.5.1.1): Normal
                Interface inside (192.168.1.1): Normal
        Other host: Secondary - Standby
                Active time: 0 (sec)
                Interface state (172.16.1.2): Normal
                Interface outside (10.5.1.2): Normal
                Interface inside (192.168.1.2): Normal

Stateful Failover Logical Update Statistics
        Link : intf3
        Stateful Obj     xmit       xerr       rcv        rerr
        General          3          0          3          0
        sys cmd          3          0          3          0
        up time          0          0          0          0
        xlate            0          0          0          0
        tcp conn         0          0          0          0
        udp conn         0          0          0          0
        ARP tbl          0          0          0          0
        RIP Tbl          0          0          0          0

        Logical Update Queue Information
                         Cur      Max      Total
        Recv Q:          0        1        3
        Xmit Q:          0        1        3
```

As you can see, there are some extra lines of output. These extra lines show stateful failover statistics in great detail.

Monitoring Failover

The primary method of monitoring failover activity is to use the *show failover* command, which can be run on either firewall. This command tells you virtually everything you want to know about failover. One of the most important pieces of information this command reveals is the status of the failover cable, which is provided in the second line of the output. It can have four possible values:

- **Normal** This means that the cable is operating normally and that the primary and secondary firewall are connected properly.
- **My side not connected** This means that the serial cable is not connected to the firewall (primary or secondary) on which you entered the command.
- **Other side is not connected** This means that the serial cable is not connected to the other firewall (the one other than the one on which you are typing this command).
- **Other side powered off** This means that the serial cable is connected to the other unit properly, but the other firewall is powered off.

In the command output, you will also see flags next to each interface. The meaning of each flag is listed here:

- **Normal** The interface is functioning properly.
- **Link Down** The line protocol on the interface is down.
- **Failed** The interface has failed.
- **Shut Down** The interface has been administratively shut down.
- **Unknown** This interface has not yet been configured with an IP address. The status of this interface has not yet been determined.
- **Waiting** Monitoring of this interface on the other firewall has not yet started.

With stateful failover enabled, the *show failover* command also displays the logical update statistics. The protocol that updates state information from the active firewall to the standby firewall over the dedicated stateful failover LAN link is known as the Logical Update (LU) protocol. The LU protocol is a real-time, UDP-like protocol that works asynchronously in the background over IP 105.

When you use stateful failover, you will see the following stateful objects listed in the Logical Update statistics section:

- **General** The sum of all objects.
- **sys cmd** Logical system update commands, such as login.
- **up time** Uptime information that is passed from the active to the standby unit.
- **xlate** The translation table.
- **tcp conn** TCP connection information.
- **udp conn** Dynamic UDP connection information.
- **ARP tbl** Dynamic ARP table information.
- **RIP Tbl** Dynamic routing table information.

For each of these stateful objects, the following statistics are available:

- **xmit** The number of packets transmitted to the other firewall.
- **xerr** The number of errors that occurred while transmitting to the other firewall.
- **rcv** The number of received packets.
- **rerr** The number of errors that occurred while receiving packets from the other firewall.

The command also displays the number of current, maximum, and total number of packets in the Logical Update transmit (*Xmit*) and receive (*Recv*) queues.

As always, for those who are interested in monitoring failover operation at a very technical and detailed level, the PIX firewall provides *debug* commands for monitoring failover operation. The command is as follows:

```
debug fover <option>
```

Here, *option* can be any of the keywords listed in Table 8.1.

Table 8.1 Failover Debug Options

Keyword	Description
cable	Failover cable status.
fail	Failover internal exception.

Continued

Table 8.1 Continued

Keyword	Description
fmsg	Failover message.
get	IP network packet received.
ifc	Network interface status trace.
open	Failover device open.
put	IP network packet transmitted.
rx	Failover cable receive.
rxdmp	Cable recv message dump (serial console only).
rxip	IP network failover packet received.
tx	Failover cable transmit.
txdmp	Cable xmit message dump (serial console only).
txip	IP network failover packet transmit.
verify	Failover message verify.
switch	Failover switching status.

Failing Back

Once failover has occurred and the primary firewall is running in standby mode and the secondary firewall is running as the active, a failback does not automatically occur. This is because there is no reason to switch the active and standby firewalls (especially if you are not using stateful failover). When the primary firewall is repaired and the failed condition has been fixed, it does not automatically become the active firewall (unless the secondary firewall now fails!). You can force the primary firewall to become active in one of two ways:

- Use the *failover active* command on the primary firewall.
- Use the *no failover active* command on the secondary firewall.

After you use one of these commands, the primary firewall will become active. If stateful failover is enabled, all connections will be maintained and no sessions will be dropped. If, however, stateful failover was not enabled, connections will be dropped and applications will have to re-establish sessions through the firewall.

Configuring & Implementing…

Performing Manual Failover for Maintenance

Manually failing over and failing back can be very useful for performing maintenance on the PIX firewalls. For example, let's say that you want to upgrade both PIX firewalls from software version 6.1 to software version 6.2. You can start by failing over to the secondary firewall by entering the *failover active* command on it. You can then power off the primary firewall, disconnect all its network and failover cables, and power it back on. Now upgrade the software on the primary firewall. Then reconnect the primary firewall, but do not power it on yet. At this point, you need to take some minimal downtime. Power off the secondary firewall, and immediately power on the primary firewall. Now disconnect all the cables from the secondary firewall, upgrade its software, and then plug everything back in. Congratulations! You upgraded the software on both firewalls with minimal downtime!

Disabling Failover

To disable failover, use the *no* form of the *failover* command:

```
PIX1(config)# no failover
```

To verify that failover has been disabled, use the *show failover* command:

```
PIX1# show failover
Failover Off
Cable Status: My side not connected
Reconnect timeout: 0:00:00
```

If you are disabling failover permanently, it is highly recommended that you clean up your configuration by removing the other *failover* commands. In our example, enter the following commands on the primary firewall:

```
PIX1(config)# no failover ip address inside 192.168.1.2
PIX1(config)# no failover ip address outside 10.5.1.2
PIX1(config)# no failover ip address state 172.16.1.2
PIX1(config)# no failover link state
```

It would be best to erase the configuration completely from the secondary firewall.

LAN-Based Failover

PIX software version 6.2 introduced support for LAN-based failover. In LAN-based failover, instead of the serial failover cable, an Ethernet link is used to monitor the failover status and exchange failover-related information. The biggest advantage of using LAN-based failover is that it gets around the distance limitation of using the serial failover cable, which is only 6 feet long. The Ethernet link used to monitor failover status in LAN-based failover must be a dedicated LAN interface. However, if you are using stateful failover, assuming you have the capacity, the same interface can be used to exchange state information. A dedicated hub or switch or a dedicated VLAN on a switch can be used to connect the two PIX firewalls for LAN-based failover. However, a crossover Ethernet cable cannot be used.

NOTE

One disadvantage of using LAN-based failover is that loss of power on the other firewall is not detected. Therefore, unlike standard failover using the serial cable, a power-loss failure is not detected until the NIC tests fail.

Configuring and Enabling Failover

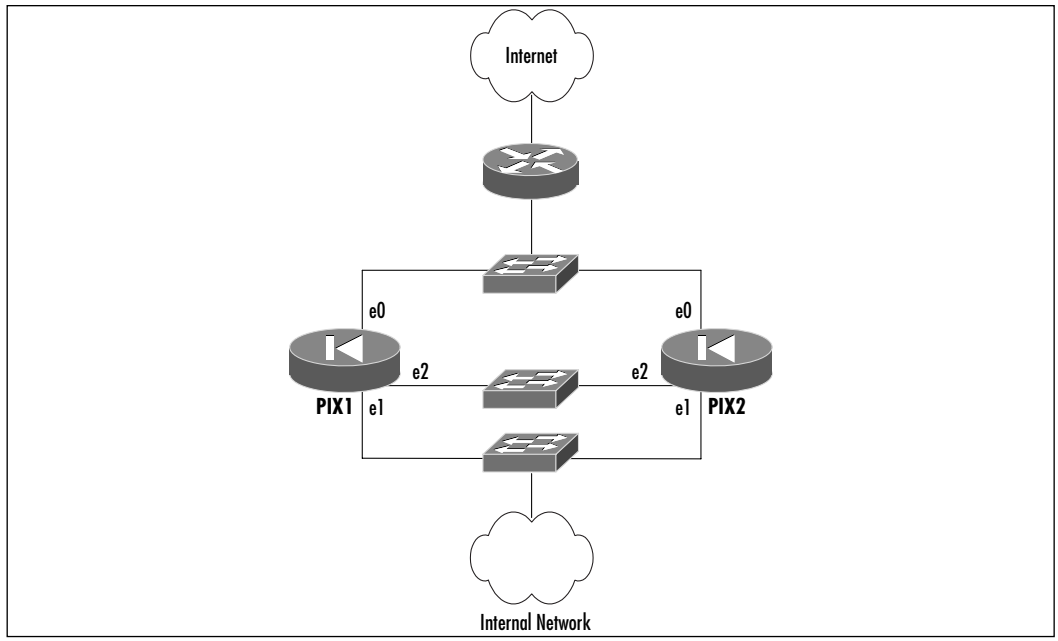
Let's use the example in Figure 8.7 to configure LAN-based failover. If a failover serial cable is connected to either of the two firewalls, you should disconnect it at this point. Connect all the network cables as shown in the diagram. We begin with the secondary firewall powered off.

As we did with failover using the serial cable, we must first set the Ethernet interface settings and assign IP addresses to each interface. By default, the inside interface (ethernet0, or e0 in the figure) and the inside interface (ethernet1, or e1) already have names assigned to them. However, ethernet2, or e2, which will be our dedicated LAN connection for failover, does not. Here is what our configuration would look like in this example:

```
PIX2(config)# nameif ethernet2 lanlink security25
PIX1(config)# interface ethernet0 100full
PIX1(config)# interface ethernet1 100full
```

```
PIX1(config)# interface ethernet2 100full
PIX1(config)# ip address inside 192.168.1.1 255.255.255.0
PIX1(config)# ip address outside 10.5.1.1 255.255.255.0
PIX1(config)# ip address lanlink 172.16.1.1 255.255.255.0
```

Figure 8.7 A LAN-Based Failover Example



First we enable failover on the primary unit:

```
PIX1(config)# failover
```

Next we configure the failover IP addresses using the *failover ip address* command:

```
PIX1(config)# failover ip address inside 192.168.1.2
PIX1(config)# failover ip address outside 10.5.1.2
PIX1(config)# failover ip address lanlink 172.16.1.2
```

We can use the *show failover* command to verify the status of the failover IP addresses (see Figure 8.8).

Figure 8.8 Output of the show failover Command After Configuring Failover IP Addresses

```
PIX1# show failover
Failover On
Cable status: Other side powered off
Reconnect timeout 0:00:00
Poll frequency 15 seconds
        This host: primary - Active
                  Active time: 300 (sec)
                  Interface lanlink (172.16.1.1): Normal (Waiting)
                  Interface outside (10.5.1.1): Normal (Waiting)
                  Interface inside (192.168.1.1): Normal (Waiting)
        Other host: secondary - Standby
                  Active time: 0 (sec)
                  Interface lanlink (172.16.1.2): Unknown (Waiting)
                  Interface outside (10.5.1.2): Unknown (Waiting)
                  Interface inside (192.168.1.2): Unknown (Waiting)
```

To designate the primary firewall for LAN-based failover, enter the following command on the primary firewall:

```
PIX1(config)# failover lan unit primary
```

We must now specify the interface that will be used to as the failover interface. On both the primary and secondary firewalls, the following command is used to select the interface:

```
failover lan interface <if_name>
```

In this example, we enter the following command on the primary firewall:

```
PIX1(config)# failover lan interface lanlink
```

In LAN-based failover, failover messages are transmitted on Ethernet links. Since these Ethernet links could be less secure than a serial cable, a manual preshared key can be used to encrypt and authenticate the contents of these messages. Although not required, it is highly recommended that you use a shared key when using LAN-based failover. The shared key is configured by typing the following command on the firewall:

```
failover lan key <secret_key>
```

In our case, we enter the following command on the primary firewall and set the key to *cisco*:

```
PIX1(config)# failover lan key cisco
```

To enable LAN-based failover on the primary firewall, enter the following commands:

```
PIX1(config)# failover lan enable
PIX1(config)# failover
```

At this point, you can power on the secondary firewall (after disconnecting the LAN-based failover interface). Enter the following commands:

```
PIX2(config)# interface ethernet2 100full
PIX2(config)# nameif ethernet2 lanlink security25
PIX2(config)# ip address lanlink 172.16.1.1 255.255.255.0
PIX2(config)# failover ip address lanlink 172.16.1.2
PIX2(config)# failover lan unit secondary
PIX2(config)# failover lan interface lanlink
PIX2(config)# failover lan key cisco
PIX2(config)# failover lan enable
PIX2(config)# failover
```

At this point, LAN-based failover is fully configured. Now you can reconnect the LAN-based failover interface. You should see the following messages on the secondary PIX firewall:

```
LAN-based Failover: trying to contact peer??
LAN-based Failover: Send hello msg and start failover monitoring
```

On the primary PIX firewall, you will see the following messages:

```
LAN-based Failover: Peer is UP
Sync Started
Sync Completed
```

If all connections are working and the configurations were typed in correctly, the *show failover* command will show that failover is operational (see Figure 8.9).

Figure 8.9 Output of the show failover Command After Completing the Configuration

```
PIX1# show failover
Failover On
Cable status: My side not connected
Reconnect timeout 0:00:00
Poll frequency 15 seconds
        This host: Primary - Active
                Active time: 400 (sec)
                Interface state (172.16.2.1): Normal
                Interface outside (10.5.1.1): Normal
                Interface inside (192.168.1.1): Normal
        Other host: Secondary - Standby
                Active time: 0 (sec)
                Interface state (172.16.2.2): Normal
                Interface outside (10.5.1.2): Normal
                Interface inside (192.168.1.2): Normal

LAN-based Failover is Active
        interface lanlink (172.16.1.1): Normal, peer (172.16.1.2): Normal
```

NOTE

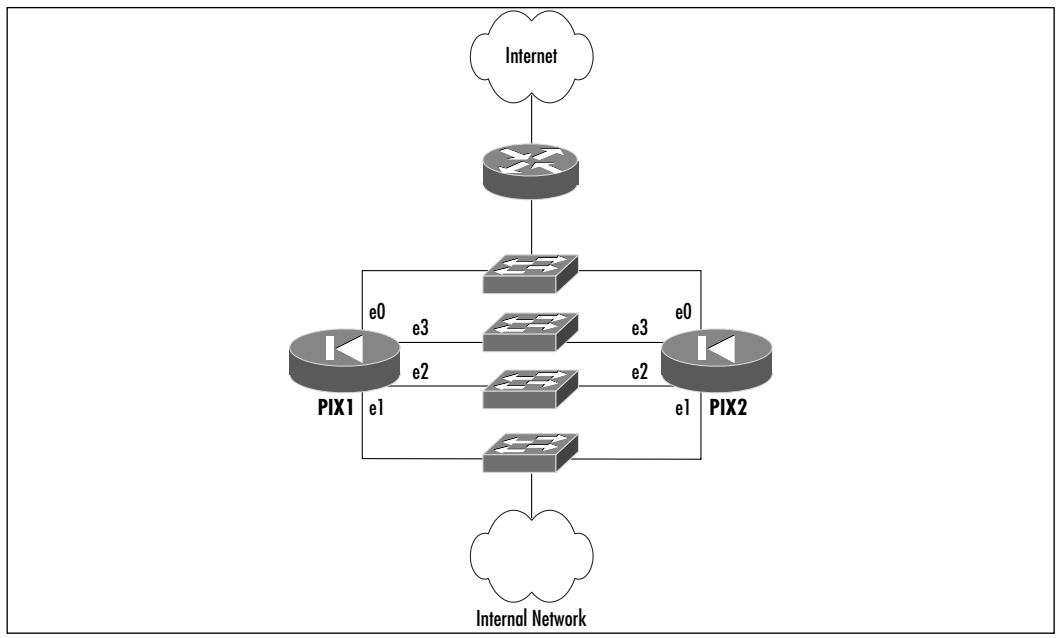
The *failover mac address* command is not available when you use LAN-based failover.

We can enable stateful failover quite easily. We will add interface ethernet3 for exchanging state information (see Figure 8.10) and configure it for stateful failover:

```
PIX1(config)# interface ethernet3 100full
PIX1(config)# nameif ethernet3 state security20
PIX1(config)# ip address state 172.16.2.1 255.255.255.0
PIX1(config)# failover ip address state 172.16.2.2
PIX1(config)# failover link state
```

```
PIX2(config)# interface ethernet3 100full
PIX2(config)# nameif ethernet3 state security20
```

Figure 8.10 A LAN-Based Stateful Failover Example



As usual, we can use the *show failover* command to check the status of stateful failover (see Figure 8.11).

Figure 8.11 Output of the show failover Command After Enabling Stateful Failover

```
PIX1# show failover
Failover On
Cable status: My side not connected
Reconnect timeout 0:00:00
Poll frequency 15 seconds
        This host: Primary - Active
                Active time: 400 (sec)
                Interface state (172.16.2.1): Normal
                Interface outside (10.5.1.1): Normal
                Interface inside (192.168.1.1): Normal
```

Continued

Figure 8.11 Continued

```
        Other host: Secondary - Standby
                Active time: 0 (sec)
                Interface state (172.16.2.2): Normal
                Interface outside (10.5.1.2): Normal
                Interface inside (192.168.1.2): Normal

Stateful Failover Logical Update Statistics
        Link : state
        Stateful Obj     xmit       xerr       rcv        rerr
        General          12         0          12         0
        sys cmd          12         0          12         0
        up time          0          0          0          0
        xlate            0          0          0          0
        tcp conn         0          0          0          0
        udp conn         0          0          0          0
        ARP tbl          0          0          0          0
        RIP Tbl          0          0          0          0

        Logical Update Queue Information
                         Cur     Max     Total
        Recv Q:          0       1       13
        Xmit Q:          0       1       13

LAN-based Failover is Active
        interface lanlink (172.16.1.1): Normal, peer (172.16.1.2): Normal
```

Monitoring Failover

Just as with standard failover using the serial cable, you can view failover status using the *show failover* command. In addition, you can get a quick status of LAN-based failover using the following command:

```
PIX1# show failover lan
LAN-based Failover is Active
        interface fail (10.20.1.1): Normal, peer (10.20.1.2): Normal
```

To view LAN-based failover details, use the *show failover lan detail* command (see Figure 8.12).

Figure 8.12 Output of the show failover lan detail Command

```
PIX1# show failover lan detail
LAN-based Failover is Active
This PIX is Primary
Command Interface is lanlink
My Command Interface IP is 172.16.2.1
Peer Command Interface IP is 172.16.2.2
My interface status is Normal
Peer interface status is Normal
Peer interface down time is 0x0

Total cmd msgs sent: 111, rcvd: 107, dropped: 0, retrans: 0, send_err: 0
Total secure msgs sent: 0, rcvd: 0
bad_signature: 2, bad_authen: 0, bad_hdr: 0, bad_osversion: 0,
    bad_length: 0
Total failed retx lck cnt: 0
Total/Cur/Max of 87:0:1 msgs on retransQ, 87 ack msgs
Cur/Max of 0:21 msgs on txq
Cur/Max of 0:1 msgs on rxq
Number of blk allocation failure: 0, cmd failure: 0, Flapping: 0
Current cmd window: 1, Slow cmd Ifc cnt: 0
Cmd Link down: 0, down and up: 0, Window Limit: 141
Number of fmsg allocation failure: 0, duplicate msgs: 0
Cmd Response Time History stat:
< 100ms:          84
100 - 250ms:      0
250 - 500ms:      0
500 - 750ms:      0
750 - 1000ms:     0
1000 - 2000ms:    0
2000 - 4000ms:    0
> 4000ms:         0
Cmd Response Retry History stat:
Retry 0 = 87, 1 = 0, 2 = 0, 3 = 0, 4 = 0
```

Continued

Figure 8.12 Continued

```
Failover enable state is 0x1
Failover state is 0x7d
Failover peer state is 0x58
Failover switching state is 0x0
Failover config syncing is not in progress
Failover poll cnt is 0
Failover Fmsg cnt is 0
Failover OS version is 6.2(2)
failover interface 0, tst_mystat = 0x0, tst_peerstat = 0x0
     zcnt = 0, hcnt = 1, my_rcnt = 10186, peer_rcnt = 23408
     myflag = 0x1, peer_flag=0x0, dchp = 0x80791f90
     act_ip: 10.5.1.171, stn_ip:10.5.1.2
     act_mac: 00d0.b7b2.97ee, stb_mac: 0090.273a.1240
failover interface 1, tst_mystat = 0x0, tst_peerstat = 0x0
     zcnt = 0, hcnt = 1, my_rcnt = 26191, peer_rcnt = 39296
     myflag = 0x1, peer_flag=0x0, dchp = 0x80791ff0
     act_ip: 192.168.1.1, stn_ip:192.168.1.2
     act_mac: 00d0.b783.9a79, stb_mac: 0090.273a.1288
failover interface 3, tst_mystat = 0x0, tst_peerstat = 0x2
     zcnt = 0, hcnt = 0, my_rcnt = 539, peer_rcnt = 404
     myflag = 0x0, peer_flag=0x0, dchp = 0x80791e10
     act_ip: 172.16.1.1, stn_ip:172.16.1.2
     act_mac: 00a0.c9ef.cfa0, stb_mac: 00a0.c9ef.cfa0
LAN-based Failover command link
```

Four new *debug* options are available (with the *debug fover* <option> command) when you use LAN-based failover: *lanrx*, *lanretx*, *lantx*, and *lancmd*. See Table 8.2 for details.

Table 8.2 LAN-Based Failover Debug Options

Option	Description
lanrx	LAN-based failover receive.
lanretx	LAN-based failover retransmit.
lantx	LAN-based failover transmit.
lancmd	LAN-based failover main thread.

Failing Back

Just as with standard failover, the *failover active* and *no failover active* commands can be used to manually change the active and standby firewalls.

Disabling Failover

The process for disabling LAN-based failover is the same as disabling standard failover. Enter the single command:

```
PIX1(config)# no failover
```

To verify that failover has been disabled, use the *show failover* command:

```
PIX1# show failover
Failover Off
Cable Status: My side not connected
Reconnect timeout: 0:00:00
```

If you are disabling failover permanently, it is highly recommended that you clean up your configuration by removing the other failover commands.

Summary

In this chapter, we learned how failover operation works on the PIX firewall. To support high availability, the Cisco PIX firewall provides the ability to deal with firewall failures. This is accomplished by running a second PIX firewall that automatically takes over in case the active firewall fails. Failover works with two, and exactly two, identical firewalls. When the active firewall fails, the standby takes over its functions.

There are two types of failover—standard failover and LAN-based failover—and the two function in a similar manner. The primary difference between them is the means of connectivity used between the primary and secondary firewalls to exchange failover information. In standard failover, a special serial cable is used to connect the two firewalls. This cable is known as the *failover cable*. In LAN-based failover, instead of the failover serial cable, an Ethernet cable is used to connect the firewalls. Whichever type of failover is used, the PIX can be set up to operate in stateless or stateful mode. Failover is only supported on the high-end models of the PIX firewall, such as the PIX 515, 515E, 520, 525, and 535. It is not supported on the PIX 501, 506, and 506E.

When you configure failover, you designate one firewall as primary and the other as secondary. In normal mode of operation, when everything is functioning properly, the primary firewall is active and handles all the network traffic. The secondary firewall sits in standby state and is ready to take over the functions of the primary firewall in case the primary fails. When the primary fails, the secondary firewall becomes active and the primary goes into the standby state. This terminology of *primary, secondary, active,* and *standby* is extremely important to understand as it relates to failover concepts.

Solutions Fast Track

Failover Concepts

- ☑ Failover works with two, and exactly two, identical firewalls. When one of these firewalls fails, the other one takes over its functions.
- ☑ Failover is only supported on the high-end models of the PIX firewall, such as the PIX 515, 515E, 520, 525, and 535. It is not supported on the PIX 501, 506, and 506E.

- ☑ Configuration replication is the process by which the configuration from the primary PIX firewall is replicated to the secondary firewall.
- ☑ In normal mode of operation, when everything is functioning properly, the primary firewall is active and handles all the network traffic. The secondary firewall sits in standby state and is ready to take over the functions of the primary firewall in case the primary fails. When the primary fails, the secondary firewall becomes active and the primary goes into the standby state.

Standard Failover Using a Failover Cable

- ☑ Standard failover works using a failover cable, which is used to connect the primary and secondary PIX firewalls. Communication over this cable includes hellos (keepalives), MAC address exchanges, state (active vs. standby), network link status, and configuration replication.
- ☑ When you use stateful failover, certain tables on the firewall, such as the translation and connection tables, are replicated from the active firewall to the standby firewall. This replication takes place over a dedicated interface. When a failover situation occurs, state information is maintained.
- ☑ The primary method of monitoring failover activity is to use the *show failover* command, which can be run on either firewall.
- ☑ Once failover has occurred, the primary firewall is running in standby mode and the secondary firewall is running as the active, and a failback does not automatically occur. You can force a firewall to be the active firewall by entering the *failover active* command in configuration mode.

LAN-Based Failover

- ☑ In LAN-based failover, instead of the serial failover cable, an Ethernet link can be used. The biggest advantage of using LAN-based failover is that it gets around the distance limitation of using the serial failover cable, which is only 6 feet long.
- ☑ A dedicated hub or switch or a dedicated VLAN on a switch can be used to connect the two PIX firewalls for LAN-based failover. A crossover Ethernet cable cannot be used.

- ☑ To enable LAN-based failover to be stateful, you can use the dedicated LAN-based failover link to exchange state information, or you can dedicate a separate LAN link for this purpose.
- ☑ To monitor LAN-based failover, use the *show failover lan* and *show failover lan detail* commands.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: How fast is a failure situation detected?

A: The time it takes to detect a failure depends on the failover poll interval. If the poll interval is set at the default of 15 seconds, network and failover communication errors are detected within 30 seconds, and power or cable failures are detected within 15 seconds.

Q: What is the best method to monitor failover on a daily basis? When the primary firewall fails, how will I know?

A: The PIX firewall generates syslog messages for all failover events, including any errors. The best way to monitor failover is to check syslog messages regularly. Failover messages are always sent with a severity level of 2 (critical). It is recommended that you install a syslog watching program to alert you when a failover error or switchover occurs.

Q: When configuring failover, how should I deal with unused interfaces?

A: If you are not using a particular interface, our recommendation is to shut it down administratively. If it is not administratively shut down, the PIX firewall will use the interface as a part of failover monitoring, and you will need to assign it system and failover IP addresses and connect it to the corresponding interface on the other firewall.

Q: Is it ever possible for failover to occur from the secondary firewall to the primary firewall?

A: Yes, there is a situation in which this failover might occur. Assume that you configure a primary and secondary firewall, and failover is working normally. In this case, the primary will be active, and the secondary will be standby. Now assume that there is a failure on the primary firewall. The secondary will take over as active, and the primary will become standby. Now assume you resolve the issue that caused the failure on the primary firewall. As you will remember, there is no automatic "failback" on the PIX firewall. This means that even though both firewalls will now be functioning normally, the secondary will continue to act as the active, and the primary will continue to be standby. If, in this situation, the secondary firewall fails, the primary will become active again, and the secondary will become standby.

Q: Can I place a router between the primary and secondary PIX firewall LAN-based failover connection?

A: No. You can only use a hub, a switch, or a dedicated VLAN on a switch. The LAN-based failover interfaces from both firewalls must be in the same VLAN and the same subnet. Unless your router is running in *transparent bridging* mode, it will not work.

Chapter 9

PIX Device Manager

Solutions in this chapter:

- Features, Limitations, and Requirements
- Installing, Configuring, and Launching PDM
- Configuring the PIX Firewall Using PDM
- Monitoring the PIX Firewall Using PDM
- Monitoring and Disconnecting Sessions

- ☑ Summary
- ☑ Solutions Fast Track
- ☑ Frequently Asked Questions

Introduction

So far, we have performed virtually all administration on the PIX firewall through the command-line interface, or CLI. In addition to the CLI for configuration and monitoring, the PIX firewall also supports a graphical user interface (GUI). Using PIX Device Manager (PDM), an administrator can use a Web browser to install, configure, and maintain the PIX firewall.

PDM is a Java-based GUI used to manage the Cisco PIX firewall. PDM is a software image that runs from flash memory on the PIX firewall, enabling administrative access via a Secure Sockets Layer (SSL) encrypted HTTPS session. PDM replaces PIX Firewall Manager (PFM) software, which was available in PIX software version 5.3(*x*) and some earlier versions. PDM allows firewall administrators to work from a variety of authorized workstations configured with a compliant browser and includes nearly all PIX CLI functionality. For example, using PDM, administrators can add, modify, and delete firewall rule sets, configure network address translation (NAT), or set up a VPN.

In addition to altering PIX configurations, PDM facilitates administrative monitoring of the PIX firewall through powerful graph and table displays for near-real-time insight into PIX performance.

In this chapter, you will learn how to install and enable PDM, specifically version 2.1. You will then learn how to use PDM's GUI to configure and monitor the PIX firewall.

NOTE

PDM is used for administration of a single firewall. Cisco Secure Policy Manager (CSPM) is a Cisco product that supports centralized management of multiple Cisco security devices, including firewalls, VPNs, and IDS sensors. A discussion of CSPM is beyond the scope of this book. You can get more information about the software at www.cisco.com/warp/public/cc/pd/sqsw/sqppmn.

Features, Limitations, and Requirements

PDM facilitates nearly all administrative functionality available in the PIX firewall CLI. This includes the ability to modify access, AAA, and filter rules on the firewall as well as implement and control NAT. PDM also gives firewall

administrators granular control of administrative functionality such as logging, IDS configuration, and user account maintenance while providing insight into current performance through the detailed PDM graphical monitoring functionality. A wealth of performance metrics and real-time statistics can easily be generated and viewed using PDM.

PDM includes powerful wizards such as the Setup Wizard and the VPN Wizard. Both tools guide firewall administrators through the often complex configuration of advanced features such as auto-update functionality and DHCP server setup or site-to-site and software client VPN configuration. PIX Device Manager 2.1 also supports object grouping, bidirectional NAT, LAN failover, several fixup configurations, Turbo ACLs, and command authorization. For information regarding these and many other supported features in the PDM interface, refer to the PDM 2.1 Release Notes at www.cisco.com/univercd/cc/td/doc/product/iaabu/pix/pdm/v_21/pdmrn21/pdmrn21.htm.

Cisco created PDM using Java applets embedded in the PDM image stored on the PIX firewall. These signed applets are downloaded directly from PIX flash memory to facilitate PIX administration free of cumbersome client-side software. Therefore, no special client software other than a compliant Web browser is required for the PDM client. However, there are several prerequisites for PDM to run successfully. These hardware, software, and client-side requirements for PDM are described in the following sections.

Supported PIX Firewall Hardware and Software Versions

The PDM application is new as of PIX firewall software version 6.0 and replaces the PFM software. PDM versions 2.0 and 2.1 require PIX software version 6.2. For versions 6.0 and 6.1 of PIX firewall, you must use PDM version 1.1. If you are using the Firewall Services Module (FWSM) version 1.1 on a Catalyst 6500 series switch, you need a minimum of version 2.1 of PDM.

PIX Device Requirements

PDM 2.1 is supported on all PIX 501, PIX 506/506E, PIX 515/515E, PIX 520, PIX 525, and PIX 535 platforms running PIX firewall software version 6.2 or higher, as well as the FWSM version 1.1 on the Catalyst 6500 series switch. Additionally, the PIX platform must meet the following requirements to run PDM:

- A minimum of 8MB total flash memory
- Data Encryption Standard (DES) or 3DES activation key

The DES or 3DES activation key enables SSL-based communication between the remote Java management client and the PIX device. PIX devices shipped with software version 6.0 and higher already include a DES activation key and encryption capabilities. 3DES, which enables stronger encryption capabilities, is available from Cisco as an additional license.

PIX devices shipped with software versions prior to 6.0 must be upgraded to version 6.0 or higher and configured with a DES activation key before PDM will function. PIX software can be downloaded from the Cisco Web site at www.cisco.com/cgi-bin/tablebuild.pl/pix. DES activation keys are available for free from Cisco on the company's Web site at: www.cisco.com/cgi-bin/Software/FormManager/formgenerator.pl?pid=221&fid=324.

NOTE

Check the PIX software version, memory, and DES capabilities using the *show version* command on the selected PIX firewall.

Requirements for a Host Running the PIX Device Management Client

Because Cisco created PDM using Java technology, several client workstations are capable of running the PDM client software. However, PDM will not function on Macintosh, Windows 3.1, or Windows 95 operating systems. PDM can be run from the operating systems shown in Table 9.1.

Table 9.1 PIX Device Manager Client OS Requirements

Client Operating Systems	OS Version
Solaris	Solaris 2.6 or higher running CDE or OpenWindows window manager.
Linux	Red Hat 7.0 or higher running the GNOME or KDE 2.0 desktop environment.
Windows	Windows 98, Windows NT 4.0, Windows 2000, Windows XP, or Windows ME.

When running PDM on Sun Solaris, the following minimum requirements apply:

- **Processor** SPARC Processor. PDM does not support Solaris on an x86 processor.
- **Memory** 128MB RAM.
- **Display** 800 x 600 pixel display with at least 256 colors (1024 x 768 pixel display and at least 16-bit color recommended).
- **Browser** Netscape Communicator 4.5x or 4.7x. PDM does not support Netscape 6.x and 7.x.

Running PDM on Linux, the following minimum requirements apply:

- **Memory** 64MB RAM.
- **Display** 800 x 600 pixel display with at least 256 colors (1024 x 768 pixel display and at least 16-bit color recommended).
- **Browser** Netscape Communicator 4.7x. PDM does not support Netscape 6.x and 7.x.

Running PDM on Windows, the following minimum requirements apply:

- **Processor** Intel Pentium or compatible processor running at 350MHz or higher.
- **Memory** 128MB RAM (192MB or higher recommended).
- **Display** 800 x 600 pixel display with at least 256 colors (1024 x 768 pixel display and at least 16-bit color recommended).
- **Browser** Internet Explorer 5.0 or higher, or Netscape Communicator 4.5x or 4.7x. PDM does not support Netscape 6.x and 7.x. Internet Explorer is recommended for higher performance.

To successfully launch PDM, your Web browser must have JavaScript and Java enabled and must support the Java Development Kit (JDK) 1.1.4 or higher. The browser must also support SSL connectivity. All browsers listed previously include this functionality.

The JDK enables the Java functionality on which PDM is based and can be obtained for supported platforms directly from Sun Microsystems at http://java.sun.com/j2se/downloads.html.

NOTE

The use of virus-checking software on the client could degrade PDM performance.

PIX Device Manager Limitations

PDM is capable of understanding and configuring nearly all Cisco PIX firewall CLI functionality. However, some CLI configurations cannot be used in conjunction with PDM. When any of the following are present on the PIX firewall, only the **Monitoring** tab is available in PDM:

- The *alias* command
- The *established* command
- The *aaa* command with the *match* option appearing in the configuration with other *aaa* commands that contain the *include* or *exclude* options
- Combined *access-list* and *access-group* commands with the *conduit* and/or *outbound* commands
- The same access list on multiple interfaces
- An ACL used for multiple purposes, such as in *access-group* and *aaa* commands
- An ACL used for multiple interfaces
- Any *outbound* command statement group applied to multiple interfaces
- Any *outbound* command statement group that contains the **except** option

PDM will not parse some CLI configurations. In these situations, PDM remains fully functional and will not remove or change the CLI format in any way. PIX firewall configurations that PDM will not parse are as follows:

- Access lists not applied to any interface and not applied to the *aaa* command
- A list of *outbound* commands without an associated *apply* command
- Any isakmp client configuration commands

PDM will also ignore any OSPF commands on the FWSM.

Installing, Configuring, and Launching PDM

This section of the chapter provides insight into the logical steps and procedures required to install, configure, and launch PDM. As detailed in previous sections, PDM and DES activation keys are preloaded on devices shipped with PIX firewall software version 6.0 and later. Additionally, some bundled versions of the PIX firewall, such as the PIX 501 3DES model, include a preinstalled 3DES key for additional security. If your PIX firewall was not shipped with software version 6.0 or later or you would like to upgrade your firewall to PDM version 2.1, follow the steps detailed in this section to install or upgrade the PIX firewall software to version 6.2 and PDM 2.1.

Preparing for Installation

Before attempting to use PDM 2.1 or configure a PIX device using PDM, verify that the PIX firewall software version of the device is 6.2 or later. If it is not, the software version must be upgraded and DES must be activated before PDM will function.

To verify the PIX firewall version, log into the CLI and type **show version**. The first two lines of the response should display the current PIX firewall version and indicate whether PDM is installed on the device. The following shows a PIX firewall with software version 6.2(2) and PDM version 2.1(1) installed:

```
PIX1# show version
Cisco PIX Firewall Version 6.2(2)
Cisco PIX Device Manager Version 2.1(1)
```

If the PIX firewall version is 6.2 or later and PDM 2.1(1) is installed, proceed to the section "Configuring the PIX Firewall Using PDM." If these are not installed, refer to the following steps to upgrade the PIX firewall, install the DES activation key, and install/upgrade PDM.

Installing or Upgrading PDM

As with all upgrade and installation procedures, begin by backing up all configuration data on the existing PIX firewall device that you plan to upgrade. If the PIX firewall is a production device, schedule the upgrade procedure during off-hours and notify the company users of the potential service outage. Doing so will help ensure a smooth upgrade process and will prevent complaints from the user community.

Verify that the PIX firewall meets all requirements listed previously in this chapter before starting with the upgrade and installation. Read all release notes carefully to determine whether any specific functionality has been removed or changed in the new release. Finally, be sure to have the software image of the PIX firewall version currently running on the PIX device backed up in the event that the new version upgrade fails and you must roll back. The installation procedure is generally trouble free, but best practice always dictates preparation for version rollback in the event of a failure.

NOTE

Administrators with a valid CCO login can find Cisco PIX firewall software and PDM images on the Cisco Web site at www.cisco.com/cgi-bin/tablebuild.pl/pix.

The basic steps for installing or upgrading PDM are:

1. Obtain a DES activation key.
2. Configure the PIX firewall for basic network connectivity.
3. Install a TFTP server and make it available to the PIX firewall.
4. Upgrade to the version of PIX firewall software and configure the DES activation key on the PIX device.
5. Install or upgrade PDM on the PIX device.

Let's take a closer look at each of these steps.

Obtaining a DES Activation Key

The first step in configuring PDM on a PIX firewall is obtaining a new activation key to enable DES encryption (if you do not already have one). A DES activation key is free from Cisco and required for PDM functionality. Because it could take some time for Cisco to issue the new key, it is best to start the request process before upgrading software on the PIX firewall. Use the *show version* command to obtain the PIX serial number. This number is required to request a new activation key. From a Web browser, go to www.cisco.com/cgi-bin/Software/FormManager/formgenerator.pl?pid=221&fid=324 and fill out the key request form. A Cisco representative will e-mail you the DES activation key shortly thereafter.

Configuring the PIX Firewall For Network Connectivity

To upgrade a PIX firewall and install PDM, the PIX firewall must first be capable of basic network connectivity. If the PIX firewall device is already on the network and capable of connecting to other devices, proceed to the next section and install a TFTP server:

1. Establish a connection to the console port of the PIX device and log into the CLI.
2. Enter Enable mode by typing **enable** at the console prompt.
3. Type **configure terminal** to enter Configuration mode on the PIX firewall.
4. Enter the setup dialog box by typing **setup** after entering Configuration mode.
5. Follow the setup dialog prompts and enter information for the following variables:
   - Enable password
   - Clock variables
   - IP address information
   - Hostname
   - Domain name
6. When prompted, save the information to write the configuration to memory.

When you're finished, physically attach the PIX firewall to the network and test for network connectivity using the *ping* command on the PIX firewall.

Installing a TFTP Server

After the PIX firewall is successfully configured on the network, a TFTP server must be installed to accommodate the new PIX firewall software and PDM software upload. Follow the instructions provided in Chapter 2 to install a TFTP server. If a TFTP server already exists, proceed to the next section and upgrade the PIX firewall software.

Upgrading the PIX Firewall and Configuring the DES Activation Key

Because PDM 2.1 only functions on PIX 6.2 and later, PIX devices with versions before 6.2 must be upgraded. Furthermore, the use of PDM requires the activation of DES or 3DES to facilitate a secure, encrypted management session. To enable DES, the new key requested in previous steps must be activated either during a new PIX image load using the Monitor mode method on the PIX firewall or using the *activation-key* command. The key on the PIX firewall cannot be changed using the typical *copy tftp flash* command.

To upgrade the PIX firewall software, follow the steps outlined in Chapter 2. If the PIX device is already running software version 6.2 and you simply need to install the new DES or 3DES license key, use the *activation-key* command from the CLI. Type **activation-key** in Configuration mode, followed by the appropriate activation key hexadecimal code provided by Cisco. To verify the key, use the *show activation-key* command.

Installing or Upgrading PDM on the PIX device

After the PIX firewall software is successfully upgraded to 6.2 and the DES or 3DES key is installed, PDM must be loaded into flash. As with the PIX firewall software upgrade, the installation of PDM is a potentially difficult operation. Always make backups of configuration files and software images before proceeding with the installation. Always verify that the PIX firewall meets the requirements specified for PDM. To install PDM, follow these steps:

1. From the TFTP server, log into CCO and download the PDM image. PDM can be found at www.cisco.com/cgi-bin/tablebuild.pl/pix.
2. Save the software in a location that can be accessed via TFTP. Note the name of the software image for later reference.
3. Log into the PIX CLI via SSH, Telnet, or the console.
4. Enter Enable mode by typing **enable** at the command prompt.
5. Type **copy tftp flash:pdm**.

NOTE

Use the *copy tftp flash:pdm* command to install the PDM image. Do *not* use the *copy tftp flash* command, because doing so will overwrite your PIX firewall operating system.

6. When prompted for the remote address of the host, type the IP address of the TFTP server.
7. When prompted for the source filename, type the name of the PDM software image on the TFTP server.
8. When prompted, type **yes** to proceed with the PDM installation.
9. After the installation is complete, type **show version** to verify that PDM is installed and that DES or 3DES is enabled. Output similar to the following should appear:

```
PIX1# show version

Cisco PIX Firewall Version 6.2(2)
Cisco PIX Device Manager Version 2.1(1)
<<  output omitted  >>
Licensed Features:
Failover:        Disabled
VPN-DES:         Enabled
VPN-3DES:        Disabled
<<  output omitted  >>
Serial Number: 480501351 (0x1ca20729)
Activation Key: 12345678 12345678 12345678 12345678
```

Enabling and Disabling PDM

Before using PDM, you must enable the PDM service and configure specific, authorized clients for administrative access. To enable PDM, you must first enter the following Configuration mode command:

```
PIX1(config)# http server enable
```

Configure PDM management clients using the following command:

```
http <ip_address> [<netmask>] [<interface>]
```

In this command, *ip_address* and *netmask* specify the client or network IP address and network mask that can access the PIX firewall through PDM. The network mask is assumed to be 255.255.255.255 (single host) if not specified. The *interface* parameter specifies the PIX interface name on which the management client will connect and is assumed to be the inside interface if not specified. For example:

```
PIX1(config)# http 192.168.1.0 255.255.255.0 inside
```

PDM is now enabled for any client on the inside interface, which is on the 192.168.1.0/24 network. Should you need to configure more clients, use the *http* command again.

NOTE

To allow PDM access from all clients, use the IP address of 0.0.0.0 with a network mask of 0.0.0.0.

To disable PDM, type **no http server enable** from the configure prompt. Doing so disables PDM for all clients. To disable specific clients, type:

```
no http <ip_address> <netmask> <interface>
```

In this command, all three parameters (*ip_address, netmask,* and *interface*) are required.

NOTE

The factory-based configuration on the PIX 501 and 506 models enables PDM by default for internal addresses. Additionally, the PIX 501 and 506 firewalls are configured with an inside interface address of 192.168.1.1 and a DHCP server that distributes 192.168.1.0/24 addresses.

Launching PDM

PDM management clients are only permitted from authorized IP addresses as specified previously by the *http* command. Before attempting to connect to the PIX via PDM, verify that the management workstation meets all functional requirements previously detailed. In addition, verify that the PDM management client is included in the *http* configuration statement on the PIX firewall. To verify that the client management station is configured for access to PDM, use the *show http* command on the PIX device.

Complete the following steps to connect to the PIX firewall with PDM:

1. Launch a JDK 1.1.4 capable browser on an authorized PDM management workstation and connect to the internal IP address of the PIX firewall using SSL.

NOTE

Be sure to type **https://**, not **http://**, in the URL string. PDM only allows encrypted access and will not function via an unencrypted link.

2. A Security Alert window will appear upon connecting to PDM the first time, as shown in Figure 9.1.

Figure 9.1 The Security Alert Window

3. When you're prompted to proceed, choose to accept the SSL security certificate by clicking **Yes**.
4. After you accept the security certificate, an authentication prompt appears, as shown in Figure 9.2. When prompted for authentication credentials, do not enter a username unless you have already configured individual user accounts via the PIX CLI. Enter the enable password in the password field and click **OK**.

NOTE

The PIX 501 and 506 platforms are not configured with a password by default. If you are connecting to these platforms for the first time using PDM, simply click **OK** to proceed.

Figure 9.2 The PDM Login Window

5. A Security Screen window will appear, as shown in Figure 9.3. Click **Yes**.

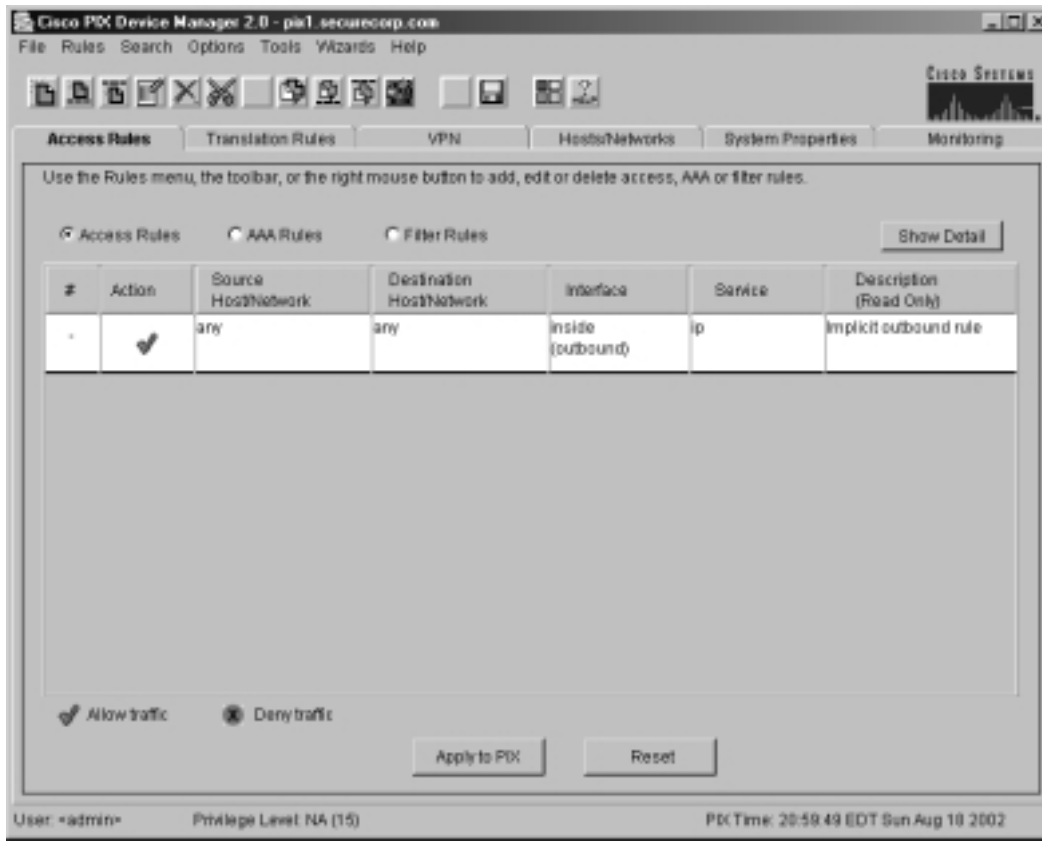
Figure 9.3 The Security Warning Window

6. PDM will launch in a separate window similar to the image shown in Figure 9.4.

From the main PDM screen, notice that there are pull-down menus, toolbar buttons, and five tabbed screens. Click the tabs and pull-down menus to become familiar with the interface. The five tabbed screens are as follows:

- **Access Rules** This screen is used to permit and deny specific network traffic traversing the PIX firewall. Additionally, AAA authentication and URL/ActiveX/Java filters are configured from the Access Rules tab.

Figure 9.4 The PIX Device Manager



- **Translation Rules** This screen is used to configure NAT pools and rules.
- **VPN** This screen is used to configure site-to-site and remote access VPNs.
- **Hosts/Networks** This screen is used to configure objects such as networks and hosts. You can also create group objects from this tab.
- **System Properties** This screen allows for basic maintenance of the PIX firewall system. Properties such as DHCP client behavior, IDS configuration, interface attributes, AAA, and other variables can be configured here.
- **Monitoring** This screen is used to monitor the PIX firewall.

In addition to the main tabbed screens available in PDM, there are several useful buttons and pull-down menus, as shown in Figure 9.5.

Figure 9.5 The PDM Main Screen and Buttons

From the File pull-down menu, you can write configuration changes to various locations such as a TFTP server or the PIX firewall as well as view and print the running configuration, refresh the PDM configuration, or reset the PIX to the factory default configuration. Cisco ships PIX firewall models 501, 506, and 506E with a factory default configuration, which is kept in the PIX firewall flash memory. The factory default configuration protects the internal network from unsolicited traffic and enables DHCP on the outside interface for automatic IP addresses acquisition. A default DHCP server address pool in the 192.168.1.0/24 network is included for hosts on the inside interface. All services are permitted outbound and translated to the firewall's external interface by default. Finally, the internal IP address of the PIX firewall is preconfigured as 192.168.1.1.

Resetting the PIX to the factory default configuration is a convenient way to erase any changes made to the configuration since it was installed and resort to an initial state of operation.

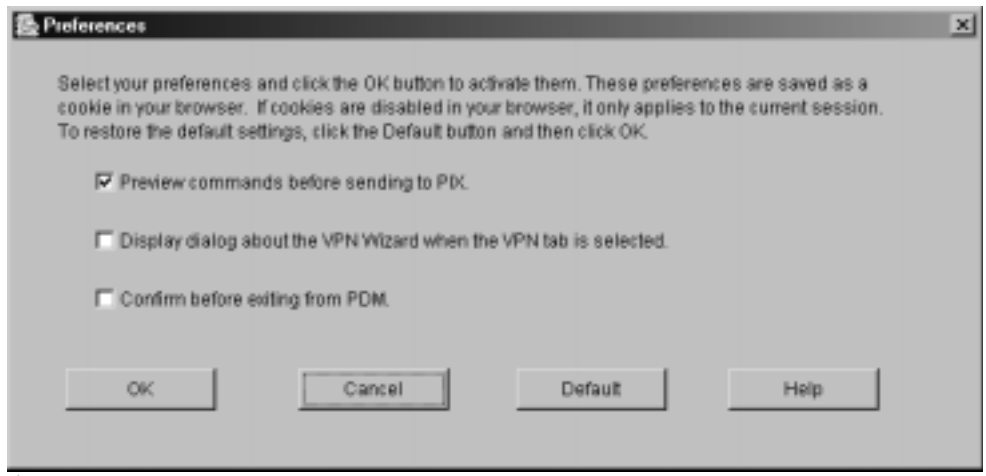
The Rules pull-down menu provides the ability to add, delete, and modify access, NAT, and VPN rules. From the Rules menu, you can also copy, cut, and paste rules for simplified maintenance. Additionally, right-click mouse functionality is available throughout the PDM interface, which mimics functionality in the Rules pull-down menu. Right-click capabilities are available wherever rules are applied to the configuration.

The Search pull-down menu provides criteria-based searching for access, NAT, and VPN rules; administrators with large and complex rule sets will appreciate this feature to quickly locate specific access rules, for instance.

From the Options pull-down menu, PDM administrators can view PIX commands not parsed by PDM, such as unapplied access lists, and can modify PDM-specific preferences from the Preferences window, as shown in Figure 9.6.

The Tools pull-down menu provides a ping utility and the ability to modify the PIX firewall directly from a Web-based CLI interface. This CLI interface can prove quite useful if you require multiline or batch firewall configuration updates. The Tools menu also provides the ability to create firewall service groups for administrators to logically group TCP and UDP services.

Figure 9.6 The Preferences Window



NOTE

When making changes to the PIX firewall via PDM, select the **Preview Commands before sending to PIX** option from the Preferences Window to learn corresponding PIX firewall CLI commands.

Two very useful features can be found under the Wizards pull-down menu: the Startup Wizard and the VPN Wizard. These wizards provide systematic prompts for the initial configuration of PIX firewall VPNs.

The final pull-down menu is Help. From Help, you will find links to detailed information regarding PDM and the PIX firewall. Help features in PDM are context sensitive. You will also find specific version information regarding PDM and the PIX firewall in this menu.

In addition, several buttons represent shortcuts to options available in the pull-down menus, such as the New Rule buttons. The New Rule buttons are represented by icons with blue pages and an orange asterisk and are intended to make rule additions quick and easy. Other buttons include the Delete Rule, Cut Rule, and Paste Rule buttons, as well as the Refresh PDM with Running Configuration and Save Running Configuration to Flash buttons.

One of the most important buttons is the Save to Flash Needed button, which appears when you have made changes to the running configuration that must be saved to the PIX flash memory. If you don't save the running

configuration to flash, any changes you make to the PIX firewall will be lost upon reboot. To learn more about a specific button, hover the mouse pointer over the button for a popup help description.

All of these tabbed screens, in addition to the pull-down menus and toolbar buttons, are discussed in detail in the following sections.

Configuring the PIX Firewall Using PDM

Configuring a PIX firewall, whether through PDM, the PIX CLI, or through Cisco Secure Policy Manager (CSPM), should be the technical application of a well-developed and well-understood security policy. Moreover, the rules implemented on the PIX firewall often represent the enforcement of the security policy. Before configuring any security device, the firewall administrator should be aware of the specific security policy of the organization. A cohesive and comprehensive technical security solution is more likely with such an approach.

Designing & Planning…

Security Policy Development

A good security practice within any organization begins with a sound and well-developed security framework. It is from this framework that policies, standards, guidelines, and standard operating procedures flow. Organizations should clearly define this framework before embarking on device configuration to ensure a uniform and predictable security stance.

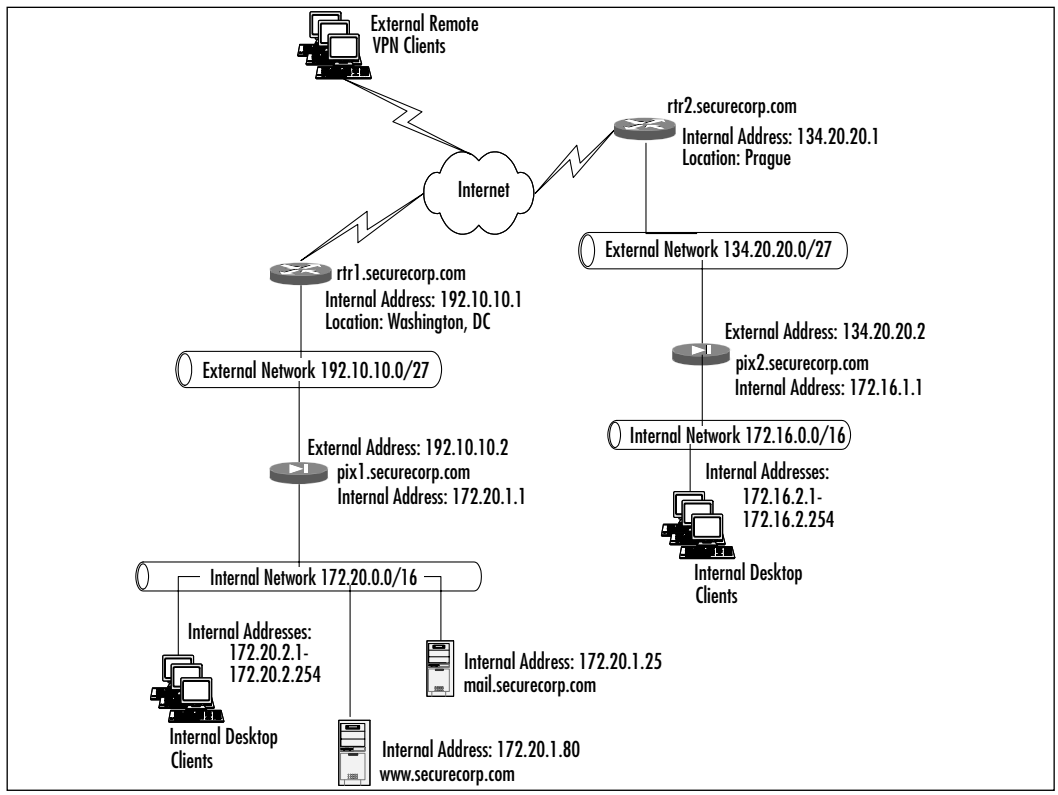
After successfully installing PDM, connect to the PIX firewall via PDM and begin configuring a specific security policy appropriate for your company. In this section, we discuss all the main tabs available in PDM and work through several exercises typical of PIX firewall implementations, such as the following:

- Using the Startup Wizard
- Configuring firewall system properties
- Implementing NAT

- Allowing inbound traffic from external sources
- Configuring VPNs

Each of these exercises is discussed in the appropriate sections in the chapter. The exercises are based on the example network architecture shown in Figure 9.7.

Figure 9.7 Our Example Network Architecture



Using the Startup Wizard

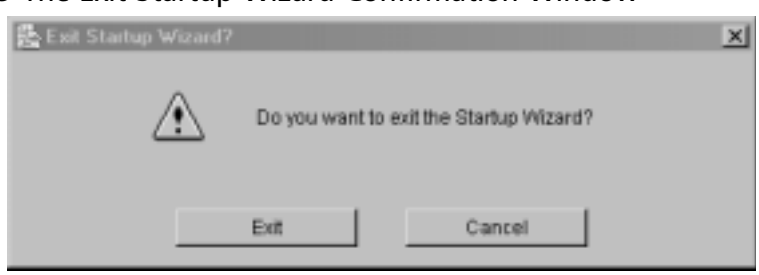
PDM includes wizards to assist firewall administrators in the initial setup and ongoing maintenance of the PIX firewall. One of these wizards, the Startup Wizard**,** guides you through typical setup configuration prompts such as interface settings, passwords, auto-update information, and others. The Startup Wizard is an excellent tool to use initially and for regular configuration changes; it extracts the current configuration and provides these PIX attributes to the administrator automatically. Therefore, the Startup Wizard process will not overwrite the current PIX firewall configuration.

This section provides a step-by-step exercise through the Startup Wizard prompts. To access the Startup Wizard, select **Startup Wizard** from the PDM Wizards menu. The Startup Wizard Welcome window appears, as shown in Figure 9.8.

Figure 9.8 The Startup Wizard Welcome Window

To proceed with the wizard, click **Next**. At any time during the Wizard process, you may exit by clicking **Cancel**. If you choose to exit the Startup Wizard, a confirmation window appears, as shown in Figure 9.9.

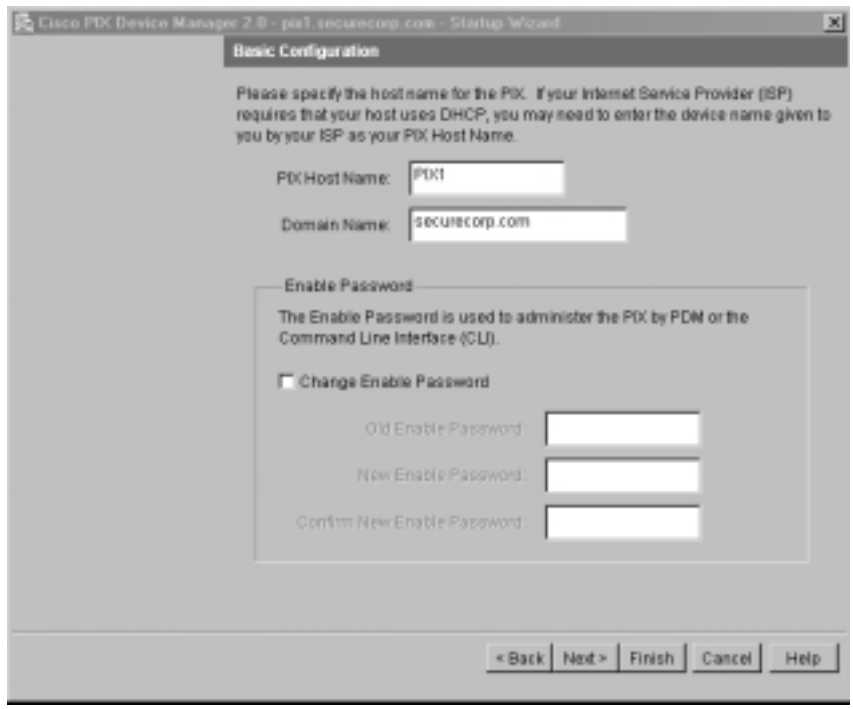
Figure 9.9 The Exit Startup Wizard Confirmation Window



To exit the Startup Window, click **Exit** and return to the PDM main window. If you want to proceed with the Startup Wizard, click **Cancel** to return to the wizard.

Click **Next** to proceed to the Basic Configuration Window. From this window, you configure the PIX hostname and domain name as well as the Enable password. The Basic Configuration Window is shown in Figure 9.10.

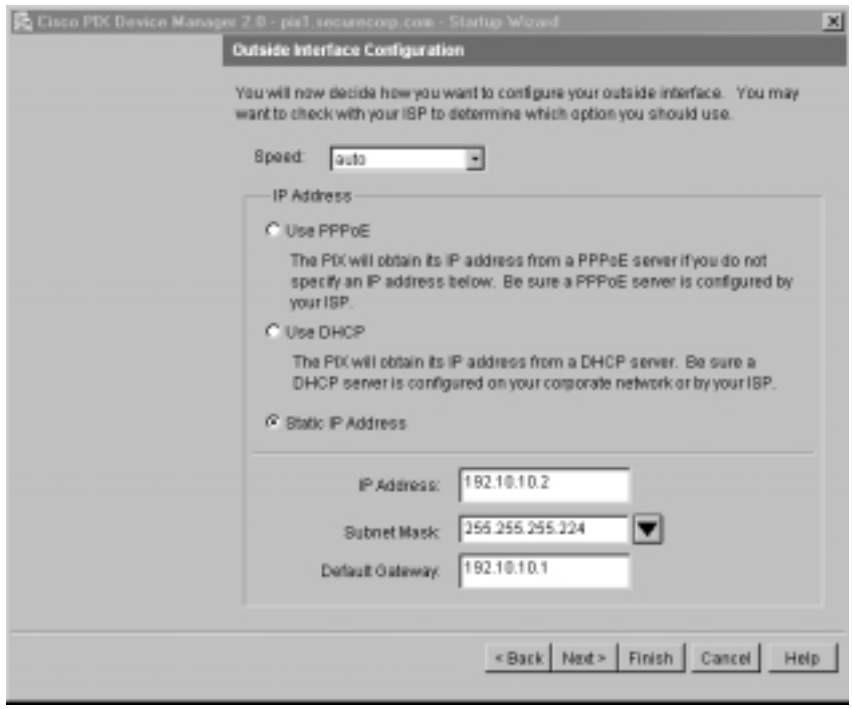
Figure 9.10 The Startup Wizard Basic Configuration Window



To change any of the settings, simply type a new hostname or domain name or click the **Change Enable Password** check box and enter new authentication credentials. You can modify these settings from the **System Properties** tab in the main PDM screen as well. To exit the Startup Wizard and save your changes at any time, click **Finish**. PDM updates the running PIX configuration and you will return to the PDM main window. To continue with the wizard, click **Next**. The Outside Interface Configuration window appears (see Figure 9.11).

From the Outside Interface Configuration window, you can select the speed of the outside interface and determine how to address the outside interface. From the wizard, you can choose to automatically configure the interface via PPPoE. You can also select DHCP to automatically determine the address of the outside interface.

Figure 9.11 The Outside Interface Configuration Window



NOTE

Before using PPPoE or DHCP to configure the outside interface, verify that your ISP is providing these services.

To statically configure the outside interface, select **Static IP Address** and provide the IP address, subnet mask, and default gateway in the field provided. To proceed with the wizard, click **Next** to set up auto-update functionality. The Auto Update Configuration window appears (see Figure 9.12).

Auto-update configuration facilitates the automated push and/or pull of PIX device configuration, PIX firewall software, and PIX PDM software data. Auto-update functionality is an advanced capability and requires externally available services to operate, but it can be extremely helpful for organizations with many PIX devices. To configure auto-update, click **Enable Auto Update** and provide the required settings. Click **Next** to proceed to the Other Interfaces Configuration window, as shown in Figure 9.13.

Figure 9.12 The Auto Update Configuration Window

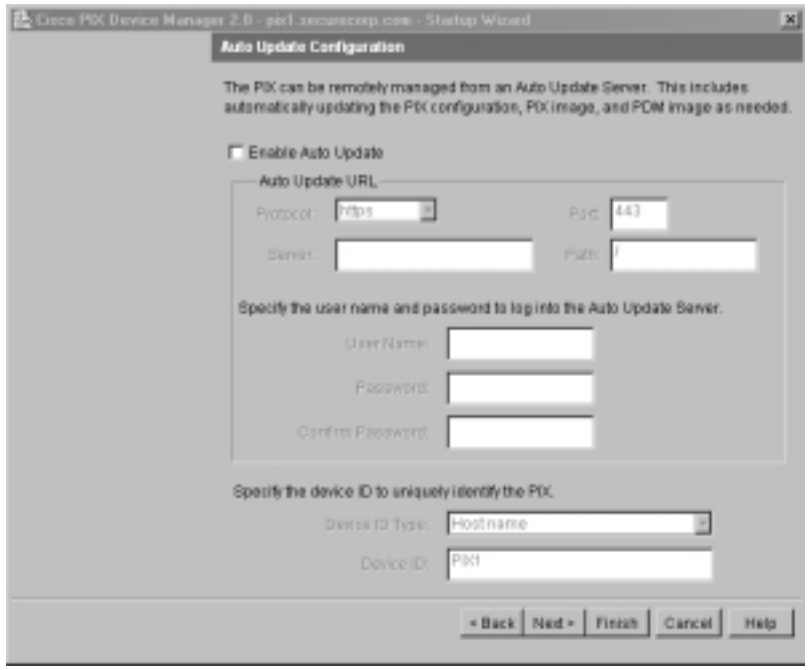
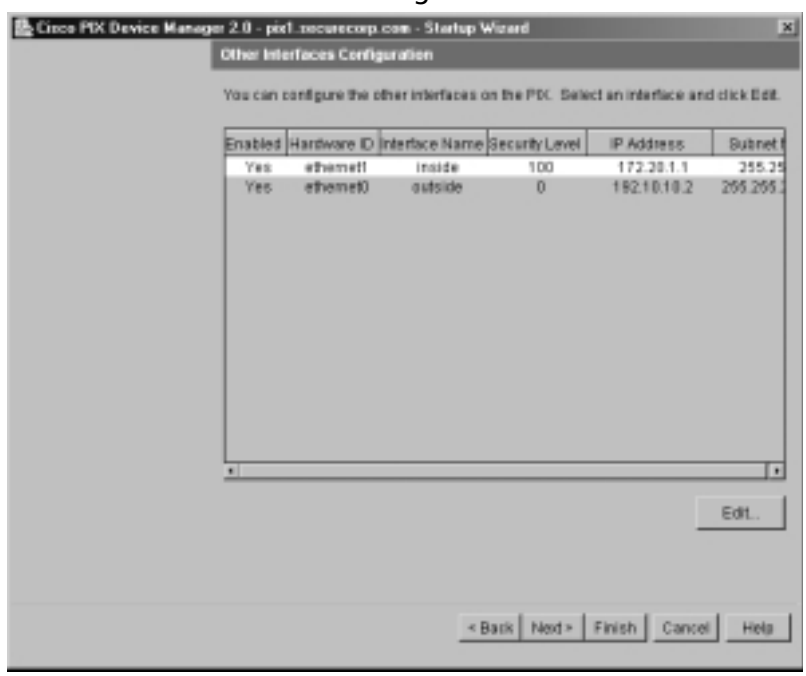
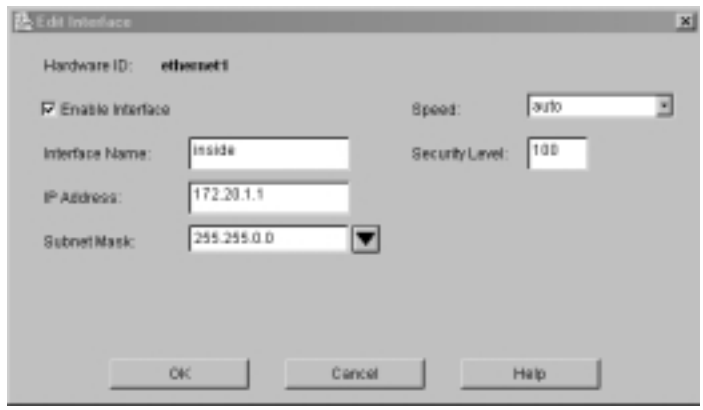


Figure 9.13 The Other Interfaces Configuration Window



From the Other Interfaces Configuration window, you can configure the remaining PIX firewall interfaces. Select an interface from the list in the Other Interfaces Configuration window and click **Edit** to change interface parameters. A window similar to that shown in Figure 9.14 appears.

Figure 9.14 The Edit Interface Window



From the Edit Interface window, you can enable or disable the interface and configure other interface parameters such as speed, security level, name, and IP address. After making configuration changes, click **OK** to return to the Other Interfaces Configuration window, then click **Next** to continue with the Startup Wizard.

The next window in the wizard is NAT and PAT Configuration. The NAT and PAT Configuration window is shown in Figure 9.15. From this window, you can configure the different types of address translation available on the PIX firewall. To configure PAT, click **Use Port Address Translation (PAT)** and either use the outside interface as the PAT address or enter a specific IP address in the space provided. If you would like to configure NAT, click **Use Network Address Translation (NAT)** and enter the appropriate global address parameters. Finally, to turn NAT off, click **Do not translate any addresses**. Click **Next** to proceed to the DHCP Server Configuration window, as shown in Figure 9.16.

The PIX firewall can act as a DHCP server for internal clients, which is quite useful in small office/home office (SOHO) environments. From the DHCP Server Configuration window, you can establish a basic DHCP server configuration. To start DHCP server operations on the firewall, click **Enable DHCP server on the inside interface** and enter a DHCP address range in the space provided. You can also alter the DHCP lease length time from the wizard as well. When you're finished, click **Next**.

Figure 9.15 The NAT and PAT Configuration Window

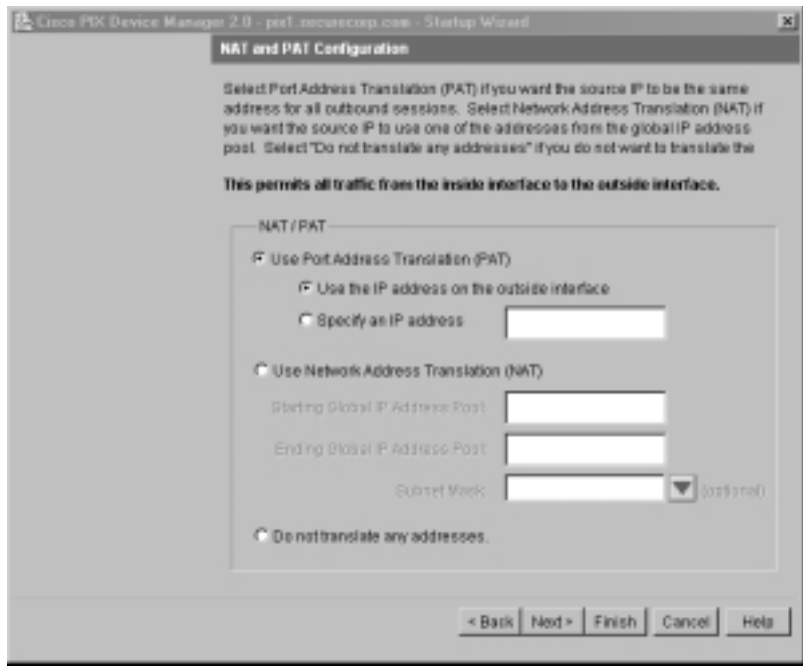
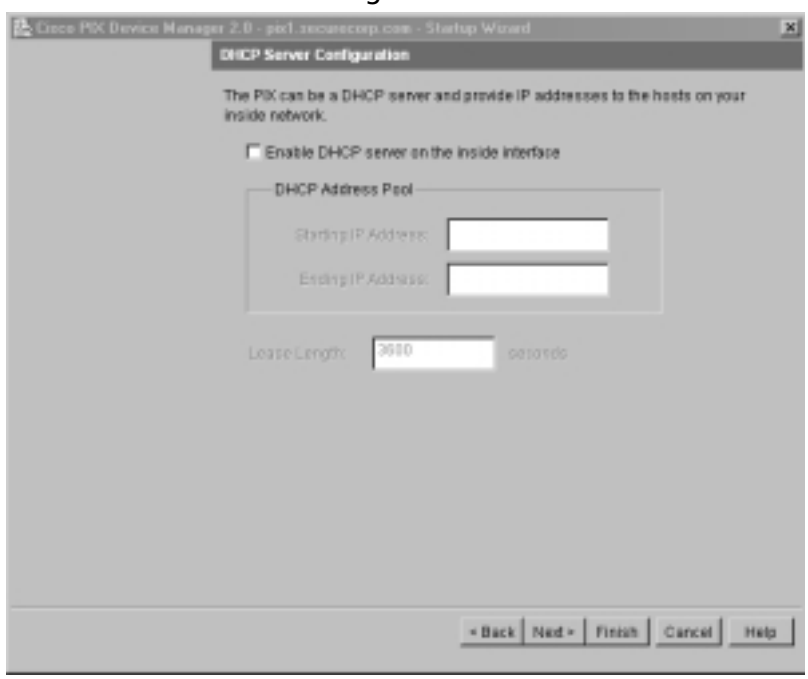


Figure 9.16 The DHCP Server Configuration Window



A screen appears to signify that the wizard is complete. Click **Finish** to exit the wizard, save the changes made during the wizard process, and return to the PDM window.

After you complete the wizard, PDM sends the updated configurations to the PIX firewall and refreshes the PIX configuration visible via the PDM interface. After making changes to the PIX firewall, you must click the **Save to Flash Needed** button to save updated configurations to the PIX flash memory. If you fail to do so, the new configurations will not be available after a reboot.

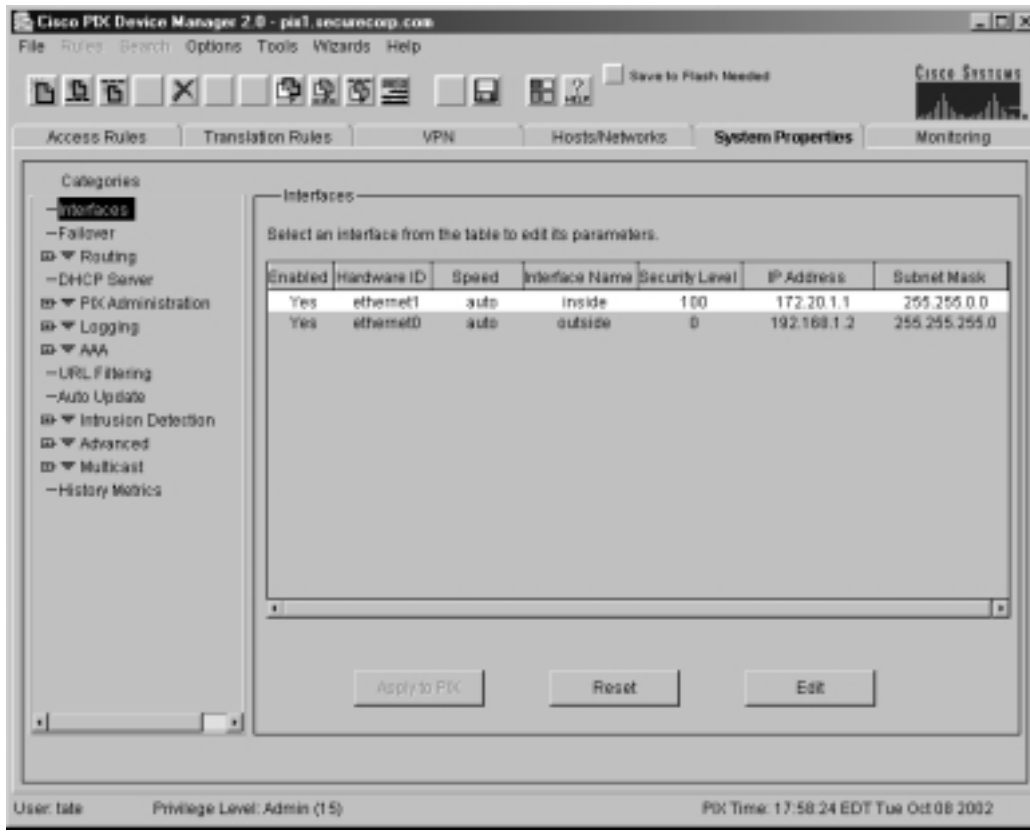
Configuring System Properties

Although the Startup Wizard is a convenient and helpful PDM utility, configuring more granular and specific properties and rules on the PIX firewall requires the use of the tabbed screens in the PDM main window. One such tab is System Properties. From this tab, you can administer many important PIX system configurations, such as the following:

- Interface properties
- Failover
- Routing
- DHCP server
- Passwords and administrative access
- Logging
- AAA
- URL filtering and auto-update
- IDS properties
- Advanced properties such as Anti-spoofing and Turbo ACLs
- Multicast

In this section, we discuss many of these categories. After connecting to the PIX firewall using PDM, click the **System Properties** tab to modify firewall properties. The System Properties screen appears, as shown in Figure 9.17.

Figure 9.17 The Interfaces Screen

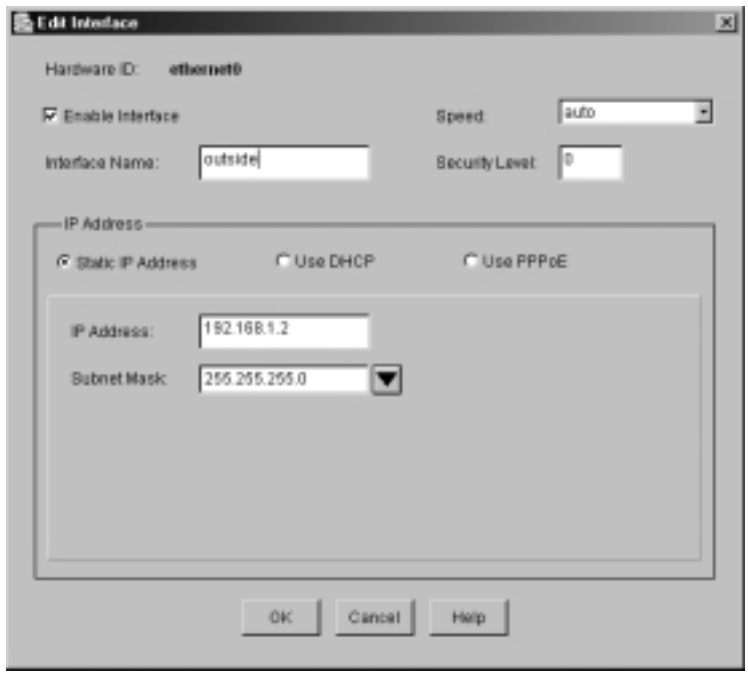


The System Properties tab is organized with a category list on the left side of the screen. When you click a specific category in this list, the right side of the screen displays configuration options appropriate for the category. Many of the categories have multiple subcomponents for configuration. To expand the category and access these subcomponents, click the small plus symbol next to the category.

The Interfaces Category

Click the **Interfaces** category listed in the left portion of the System Properties screen, as shown in Figure 9.17. From this category, you can make changes to all PIX interfaces. Highlight the specific interface you want to modify and click the **Edit** button. The Edit Interface screen appears, as shown in Figure 9.18.

Figure 9.18 The Edit Interface Window



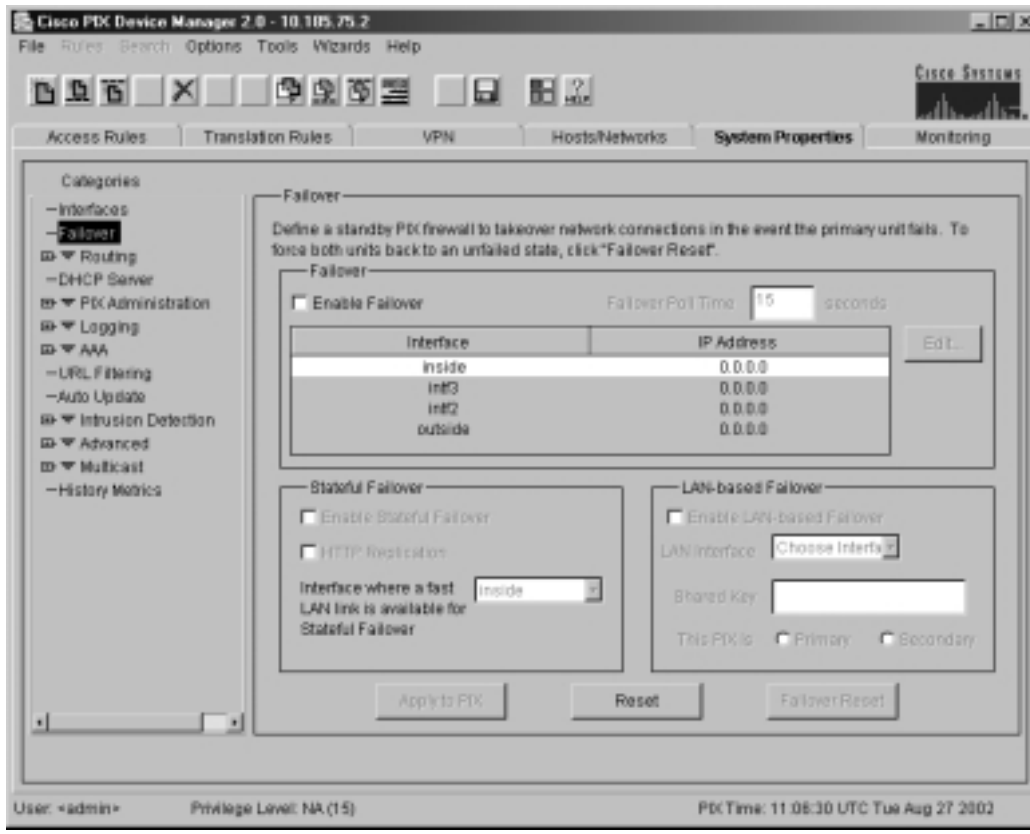
This example shows a statically configured outside interface with IP address 192.10.10.2 and a subnet mask of 255.255.255.224. Modify the interface attributes that require change and click **OK**. From the System Properties screen, click the **Apply to PIX** button to save changes to the PIX running configuration.

The Failover Category

The second category listed under System Properties is Failover. From this category, you can set up and configure failover between two identical PIX firewalls. When configuring PIX firewalls for failover, you must first configure certain attributes, such as the failover interface IP addresses. After enabling failover between two PIX firewalls, all maintenance is performed from the primary firewall in the pair. The Failover category screen is shown in Figure 9.19.

From this screen, you can enable or disable failover by clicking the check box beside **Enable Failover**. To configure the failover IP addresses for each interface, highlight a specific interface and click **Edit**. The failover IP addresses are assumed by the standby PIX during normal operations.

Figure 9.19 The Failover Screen



NOTE

You must have an appropriate failover license to access the Failover category. Without a failover license, you cannot configure failover from PDM or from the CLI.

From the Failover screen, you can also control stateful failover and LAN-based failover properties. To enable stateful failover, click the **Enable Stateful Failover** check box and select a high-speed interface from the pull-down list for stateful synchronization. The PIX firewalls will use this interface to pass connection state data. To enable HTTP replication, click the check box beside **HTTP Replication**. Doing so configures the PIX firewalls to exchange HTTP connection data across the stateful synchronization link.

LAN-based failover permits the increased physical separation of PIX devices configured in a failover pair. Previously, PIX firewalls configured in a pair required a six-meter or less separation due to serial cable limitations. To enable LAN-based failover, click the **Enable LAN-based Failover** check box and select the interface used for failover status checking from the **LAN Interface** drop-down list. In addition to selecting the LAN interface, you must choose a shared key to use between the failover PIX devices. Type the key in the **Shared Key** field; remember this key because you must apply the identical key string to the second PIX firewall. Finally, determine whether the PIX firewall will be the primary or secondary firewall during normal operations by clicking either the **Primary** or the **Secondary** radio button.

When you are finished, click the **Apply to PIX** button. At any point in this process, you can click **Reset** to return the Failover screen attributes to their original values. The Failover Reset button is used to reset the failover state.

The Routing Category

Another category available on the System Properties tab is Routing; from here, you can configure dynamic routing via RIP, add static routes to the PIX firewall, and configure proxy ARP. Here we examine the process of adding static routes to the firewall. The PIX firewall is capable of accepting RIP routing updates, but RIP is often less preferred in enterprise environments due to slow convergence times and limited scaling capabilities. To add a static default route, click the **Routing** category listed in the left portion of the System Properties screen. From the expanded category list, click **Static Route**, as shown in Figure 9.20.

Click **Add** to add a new static route. The Add Static Route window appears, as shown in Figure 9.21.

Here we are adding a default route by setting the IP Address and Mask fields to 0.0.0.0. By specifying the outside interface in the **Interface Name** drop-down list, you configure the PIX to route all traffic through the outside interface to the IP address listed in the **Gateway IP** field. Add the required route information, as shown in the following discussion, and click **OK**.

From the System Properties screen, click the **Apply to PIX** button to save the route information to the PIX running configuration.

Figure 9.20 Routing: The Static Route Screen

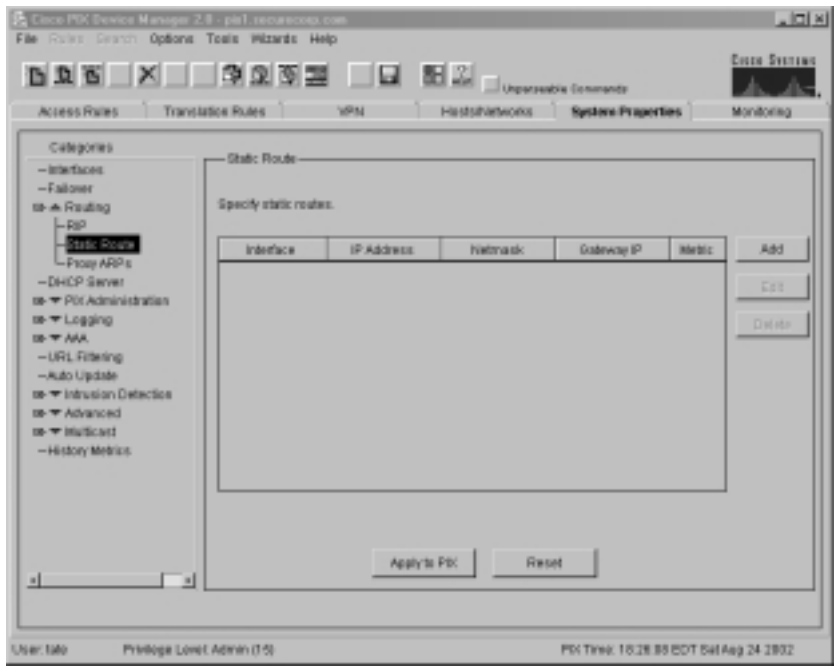
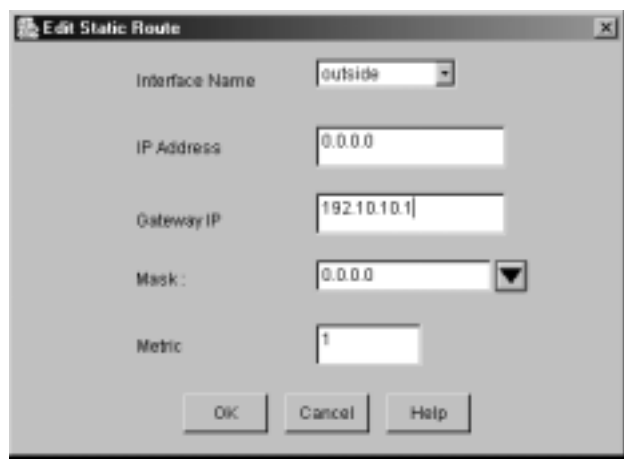


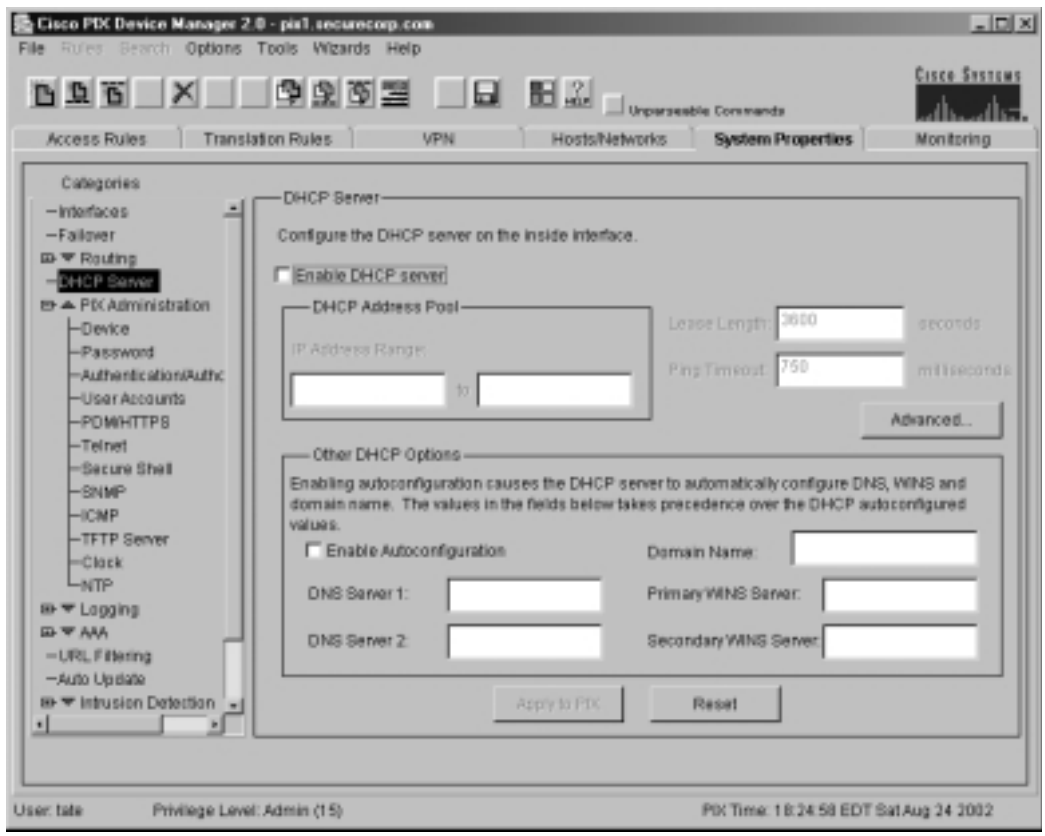
Figure 9.21 The Add Static Route Window



The DHCP Server Category

As previously mentioned, the PIX firewall can be used as a DHCP server. This is extremely beneficial in small office and home environments where access to additional server equipment could be limited. To add or alter DHCP server configurations, click the **DHCP Server** category from the System Properties tab. The DHCP Server screen appears, as shown in Figure 9.22.

Figure 9.22 The DHCP Server Screen



Enable DHCP services on the PIX firewall by clicking the **Enable DHCP server** check box. From this screen, you can specify the DHCP Address Pool and other DHCP options such as DNS Servers, Domain Name, and WINS Servers. You can also specify the DHCP Lease Length and Ping Timeout values. Click the **Advanced** button to configure IP telephony options. The DHCP Server Advanced window appears, as shown in Figure 9.23.

Figure 9.23 The DHCP Server Advanced Window

These options supply TFTP variables to devices on the network such as Cisco IP telephones so that the devices can download software.

The PIX Administration Category

Perhaps one of the most useful categories available in the System Properties tab is the PIX Administration category. From this category, you can administer the subcategories listed in Table 9.2.

Table 9.2 PIX Administration Subcategories

Pix Administration Subcategory	Function
Device	Configure the PIX hostname and domain name.
Password	Configure enable and Telnet passwords.
Authentication/ Authorization	Configure LOCAL, TACACS+, or RADIUS authentication/ authorization for PDM, serial, Telnet, and SSH connections. Specific authorization levels can also be administered from this screen.
User Accounts	Administer local user accounts and define privilege levels.
PDM/HTTPS	Specify hosts and networks allowed to access the fire wall via PDM.
Telnet	Specify hosts and networks allowed to access the firewall via Telnet.

Continued

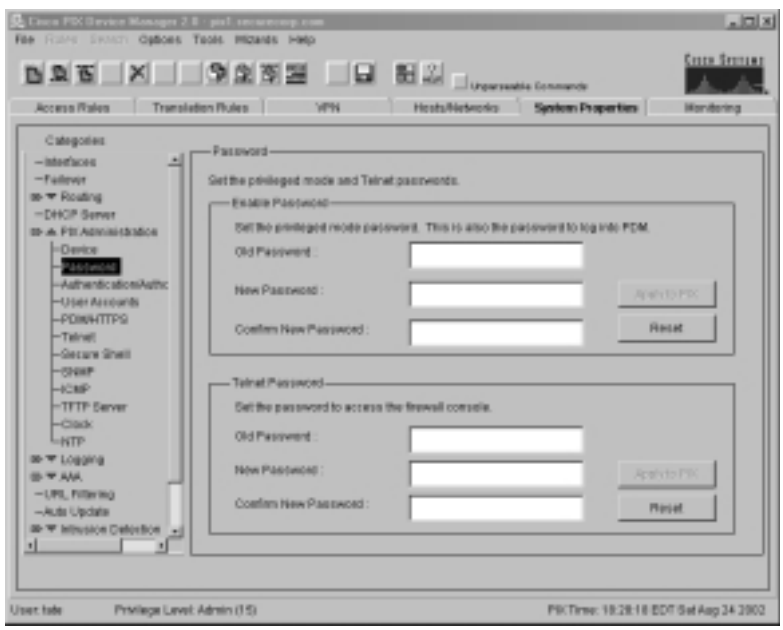
Table 9.2 Continued

Pix Administration Subcategory	Function
Secure Shell	Specify hosts and networks allowed to access the firewall via SSH.
SNMP	Configure SNMP variables such as community strings and trap destinations.
ICMP	Configure ICMP permissions to the firewall interfaces.
TFTP Server	Specify TFTP services.
Clock	Configure time variables such as time zone and calendar date.
NTP	Configure network time protocol servers for automated time synchronization.

Click each subcategory to view the related configuration screen. The Device screen is identical to that shown in the Startup Wizard window. Refer to the "Startup Wizard" section for additional information about changing the hostname and domain name for the PIX device.

To change administrative authentication variables on the PIX firewall, click the **Password** subcategory in the **PIX Administration** category, as shown in Figure 9.24.

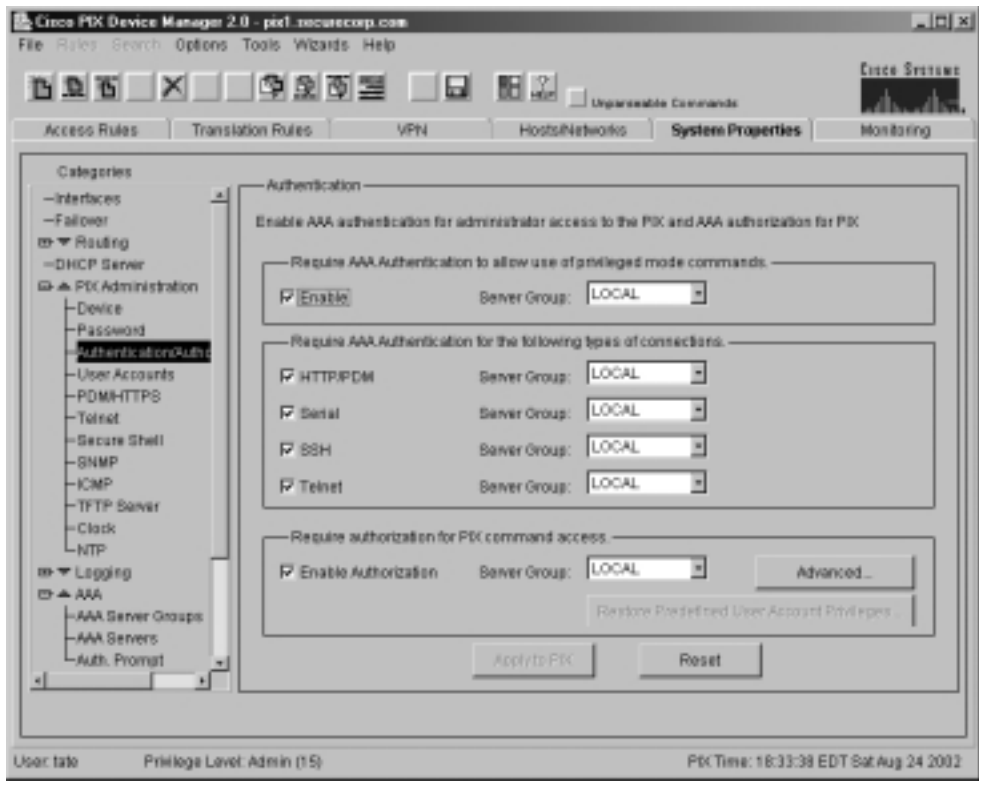
Figure 9.24 PIX Administration: The Password Screen



To change passwords, type the existing Enable or Telnet password in the **Old Password** field. Type a new password in the **New Password** and **Confirm New Password** fields. Click **Apply to PIX**. A dialog box appears confirming the new password. To reset the Enable or Telnet password to the original configuration, click **Reset**.

You can control very granular authentication and authorization attributes via the Authentication/Authorization PIX Administration subcategory screen. This screen is shown in Figure 9.25.

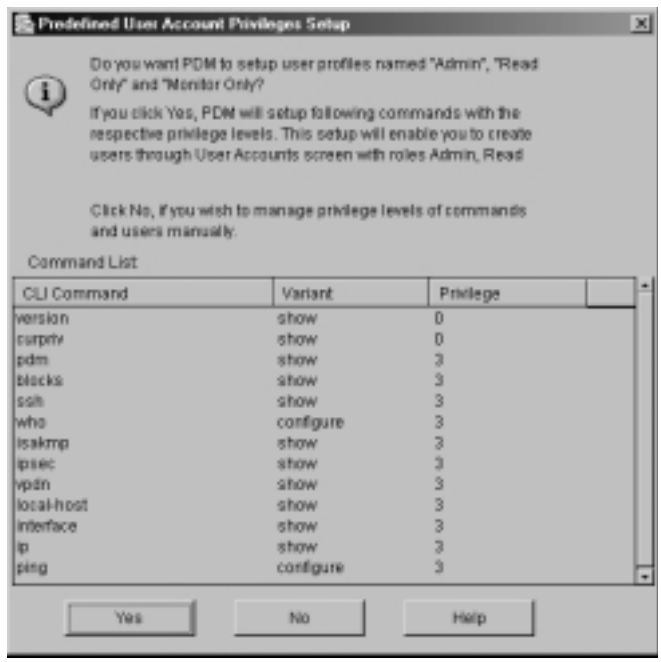
Figure 9.25 PIX Administration: The Authentication/Authorization Screen



Access to the PIX firewall, whether via PDM, serial (console), SSH, or Telnet, can be controlled via LOCAL, TACACS+, or RADIUS server groups. You can also configure authentication for privileged modes from this screen. AAA server groups are determined using the AAA category from the System Properties window. (AAA is discussed later in this chapter.) If no AAA services are available on your network, you can use LOCAL authentication and configure user accounts on the PIX firewall. User account maintenance is discussed later in this section.

Use the check boxes and pull-down menus to alter authentication and authorization attributes. From PDM, you can also control user access to administrative commands. This PDM feature enables distributed administration and allows users to access PDM using read-only or monitor-only permissions. To enable this feature, click the **Enable Authorization** check box. When you first enable authorization, PDM prompts you to configure predefined account privileges. A window appears, as shown in Figure 9.26.

Figure 9.26 The Predefined User Account Privileges Setup Window



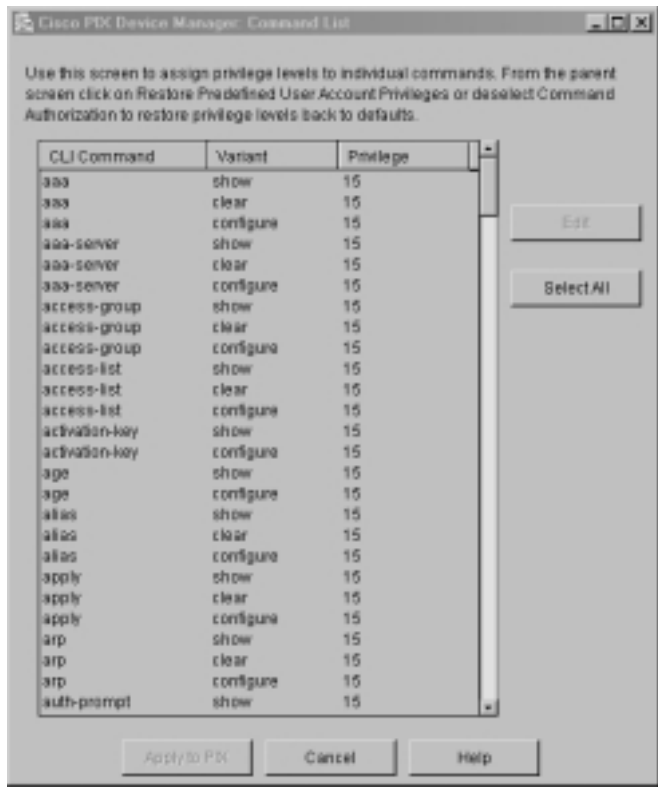
This setup screen creates three predefined authorization levels, which are detailed in Table 9.3.

Table 9.3 Predefined Authorization Levels

Predefined Authorization	CLI Level	Description
Admin	15	Access to CLI functionality.
Read-only	5	Read-only access to all CLI functionality.
Monitor-only	3	Access to monitoring functionality only.

Click **Yes** to create these predefined authorization levels or **No** to specify your own levels. To specify your own granular command access attributes, click the **Advanced** button. The Command List window appears, as shown in Figure 9.27.

Figure 9.27 The Authentication/Authorization Command List Window



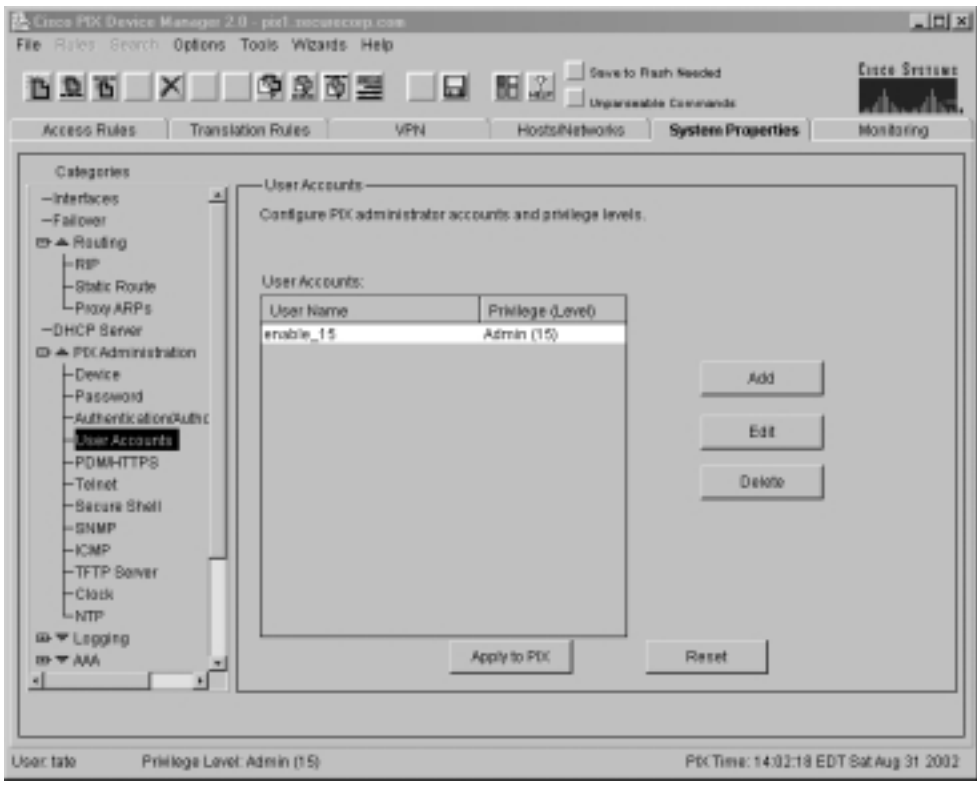
From this window, you can specify the privilege level for each PIX firewall command. To change a privilege level, highlight a CLI command and click **Edit**. From the popup window, select the privilege level from the **Privilege Level** drop-down list. When finished, click **Apply to Pix** and return to the **System Properties** tab.

After specifying authentication mechanisms and authorization levels, you can add administrative user accounts from the **User Accounts** category, as shown in Figure 9.28.

From this screen, you can add, modify, or delete user accounts. To add a new user, click the **Add** button. From the popup window, configure user attributes by completing the **User Name** and **Password** fields and selecting an appropriate

privilege level from the **Privilege Level** pull-down list. If you configured specific privilege levels as previously discussed, these levels will appear in the **Privilege Level** pull-down list.

Figure 9.28 PIX Administration: The User Accounts Screen



The next three PIX Administration subcategories are similar in nature. These subcategories, PDM/HTTPS, Telnet, and Secure Shell, all control source IP address access to each of these administration methods. For brevity, we only discuss the PDM/HTTPS subcategory here.

To add, delete, or modify the source IP addresses permitted to access PDM, click the **PDM/HTTPS** subcategory. The PDM/HTTPS screen appears, as shown in Figure 9.29.

Use the **Add**, **Edit**, and **Delete** buttons from this screen to control the IP address(s) allowed to access PDM from specific interfaces. This screen is identical for Telnet and Secure Shell access control.

From the PIX Administration category, you can also configure SNMP. To modify SNMP variables, click the **SNMP** subcategory. The SNMP screen appears, as shown in Figure 9.30.

Figure 9.29 PIX Administration: PDM/HTTPS

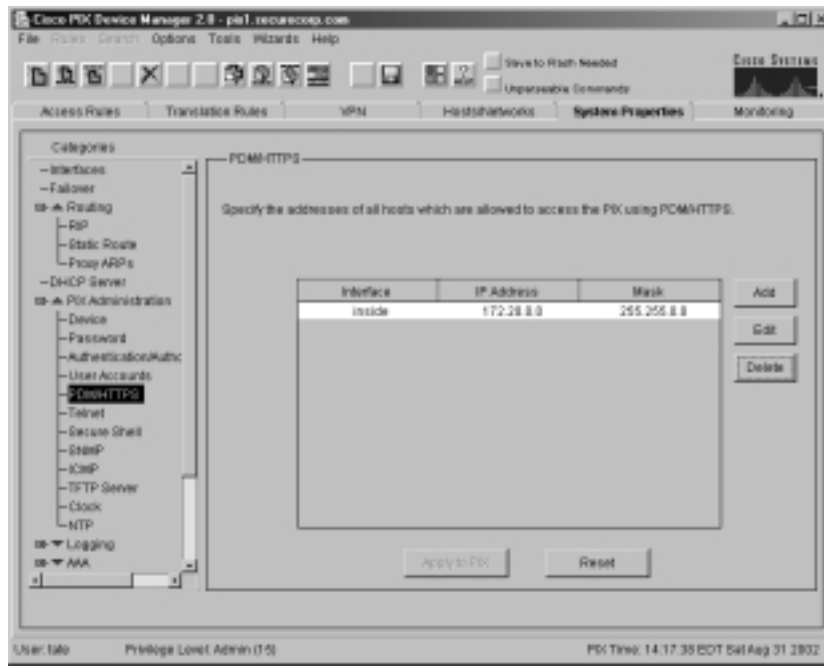
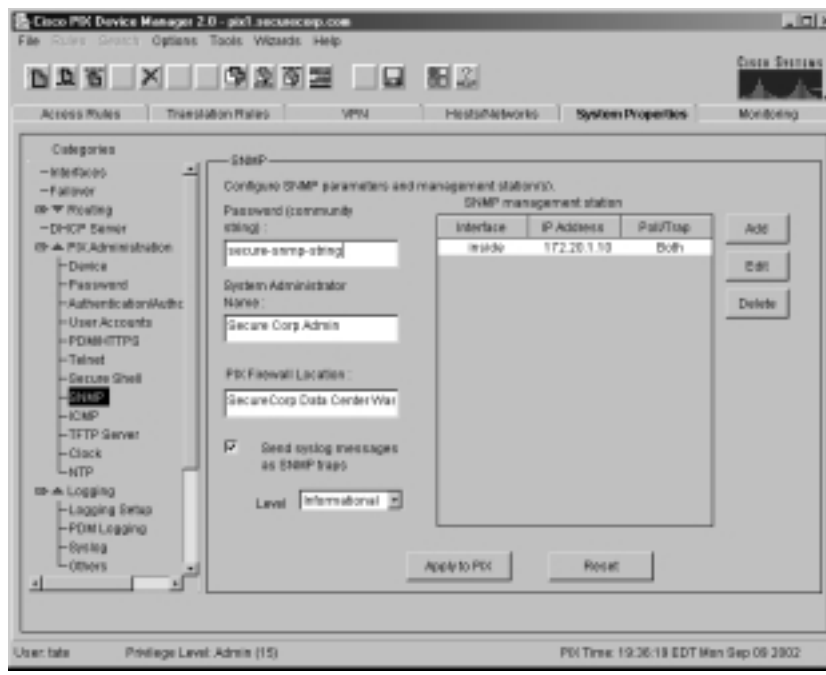


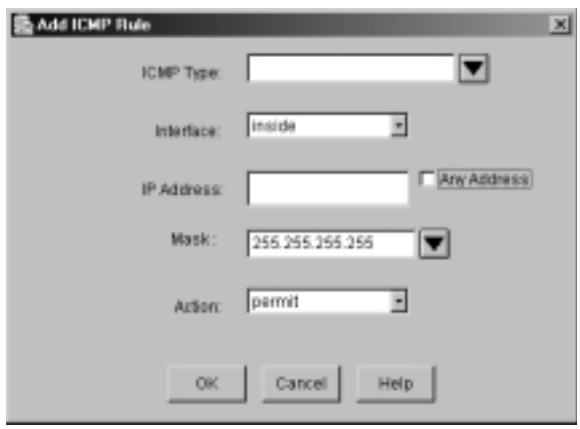
Figure 9.30 PIX Administration: SNMP



As you can see, from the SNMP screen you can configure variables such as the community string, contact, and location information. You can also specify SNMP management stations by clicking the **Add** button beside the **SNMP management station** field. When you add a management station, you can configure the PIX to allow polling from the server and to send traps to the server. To configure the firewall to send syslog-based traps to the server, click the check box beside the **Send syslog messages as SNMP traps** and select the severity level from the **Level** pull-down list.

ICMP is a useful testing and debugging tool in any environment. The ICMP subcategory is used to permit or deny ICMP to the PIX interfaces. This should not be confused with ACLs applied to the PIX interfaces to permit or deny ICMP *through* the firewall. To disable ICMP or permit only specific types of ICMP to the PIX interfaces, click the **Add** button from the ICMP screen. The Add ICMP Rule window appears, as shown in Figure 9.31.

Figure 9.31 The Add ICMP Rule Window



From this window, select the ICMP type such as **echo** or **echo-reply** from the **ICMP Type** pull-down menu, choose the interface from the **Interface** pull-down menu, and determine the source address information by typing in the IP Address and Mask fields. Finally, determine the PIX action by selecting either **permit** or **deny** from the **Action** pull-down list.

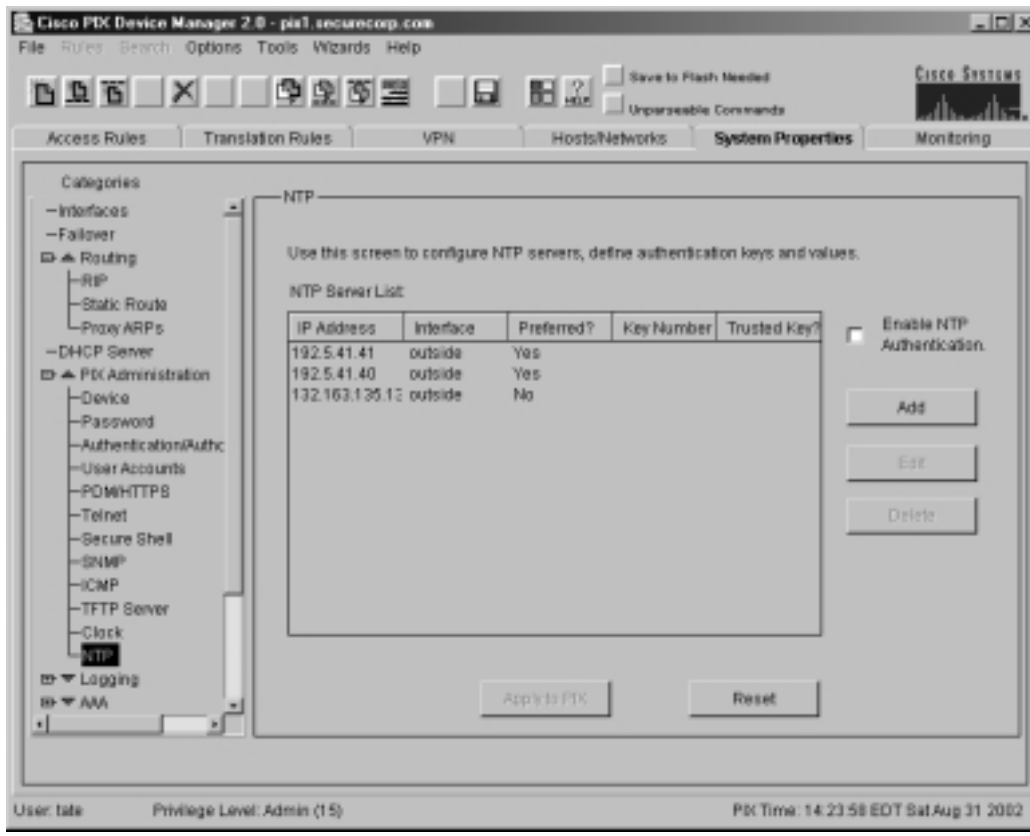
NOTE

By default, the PIX firewall permits ICMP to all its interfaces.

The TFTP Server subcategory permits the configuration of TFTP variables such as the IP address of the TFTP server and the specific server file system path to be used for TFTP transfers.

The last two subcategories under PIX Administration control the date and time on the PIX firewall. The Clock subcategory facilitates the configuration of the time zone, day, month, year, and exact time on the PIX firewall. The NTP subcategory permits the configuration of Network Time Protocol (NTP) attributes. Click the **NTP** subcategory to see the NTP screen, as shown in Figure 9.32.

Figure 9.32 PIX Administration: NTP



From this screen, you can add NTP clock sources to maintain accurate time. Use the Add, Edit, and Delete buttons to configure NTP server IP addresses and the interface over which NTP should run. Select the **Enable NTP Authentication** check box to enable NTP.

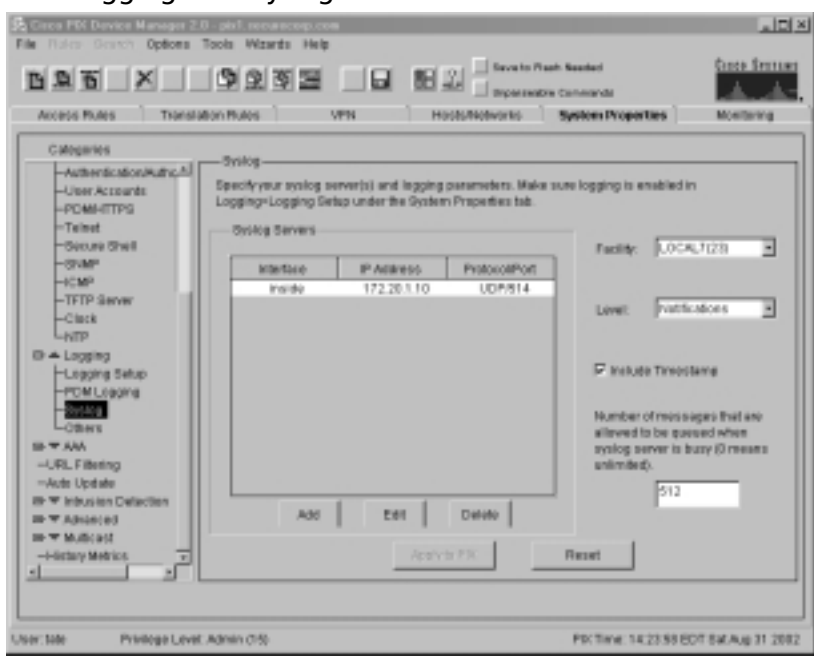
The Logging Category

The next category available under the System Properties tab is Logging. Good security practices involve logging network-related events for diagnosing problems and correlating potential security events. From the Logging category, you can enable logging, set the logging level, and specify syslog servers. You can also control PDM logging and logging to other locations such as buffered memory, console, or Telnet/SSH.

To enable logging, click the **Logging Setup** subcategory and click the **Enable Logging** check box. From this screen, you can also suppress specific logging message that could occur frequently on your firewall. If you use PDM often to manage your firewall, it is prudent to log these administrative actions for insight into changes made to the firewall. Configure PDM logging from the PDM Logging subcategory. From this subcategory, you can choose the level of logging as well as the size of the PDM logging buffer, which will determine the maximum size of the log file retained on the PIX firewall.

Typically, firewall administrators configure a syslog host on the network to aggregate logging from various devices providing security and network connectivity. From PDM, you can specify syslog attributes. To do so, click the **Syslog** subcategory. The Syslog screen appears, as shown in Figure 9.33.

Figure 9.33 Logging: The Syslog Screen



To add a syslog server, click **Add** and configure the interface, IP address, and protocol/port of the syslog server. You can also determine the logging facility as configured on the syslog server from this screen under the Facility pull-down list. From this screen, you can also select the level of logging to be sent to the syslog host. The PIX firewall can be configured to send logging information ranging from critical to debug level to a syslog server. Each logging level increases the quantity of data sent to the syslog host, so be careful when you set the syslog level.

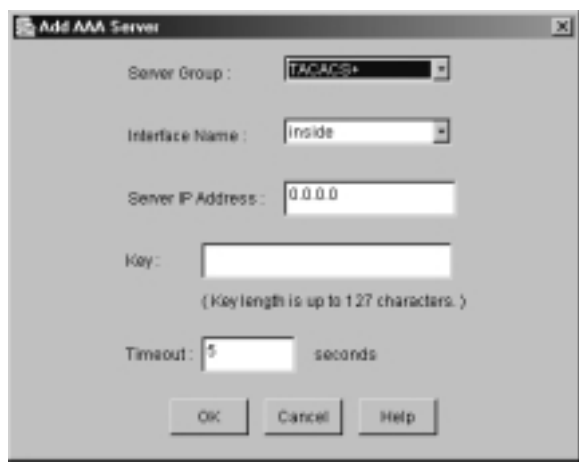
The final subcategory under the Logging category is Others. From this subcategory, you can determine whether the PIX firewall logs to other mediums such as console or Telnet sessions or to the PIX internal buffer.

The AAA Category

The AAA Category available from the System Administration tab facilitates the configuration of Cisco authentication, authorization, and accounting variables through the AAA Server Groups, AAA Servers, and Auth. Prompt subcategories. Click each of these subcategories to view the options contained therein. Three AAA server groups are predefined and visible from the AAA Server Groups subcategory, TACACS+, RADIUS, and LOCAL. These default groups can be used in your configuration, or you can add new groups by clicking the **Add** button. New groups can be either RADIUS or TACACS+ based.

To add a new AAA server, select the **AAA Server** subcategory and click **Add**. The Add AAA Server window appears, as shown in Figure 9.34.

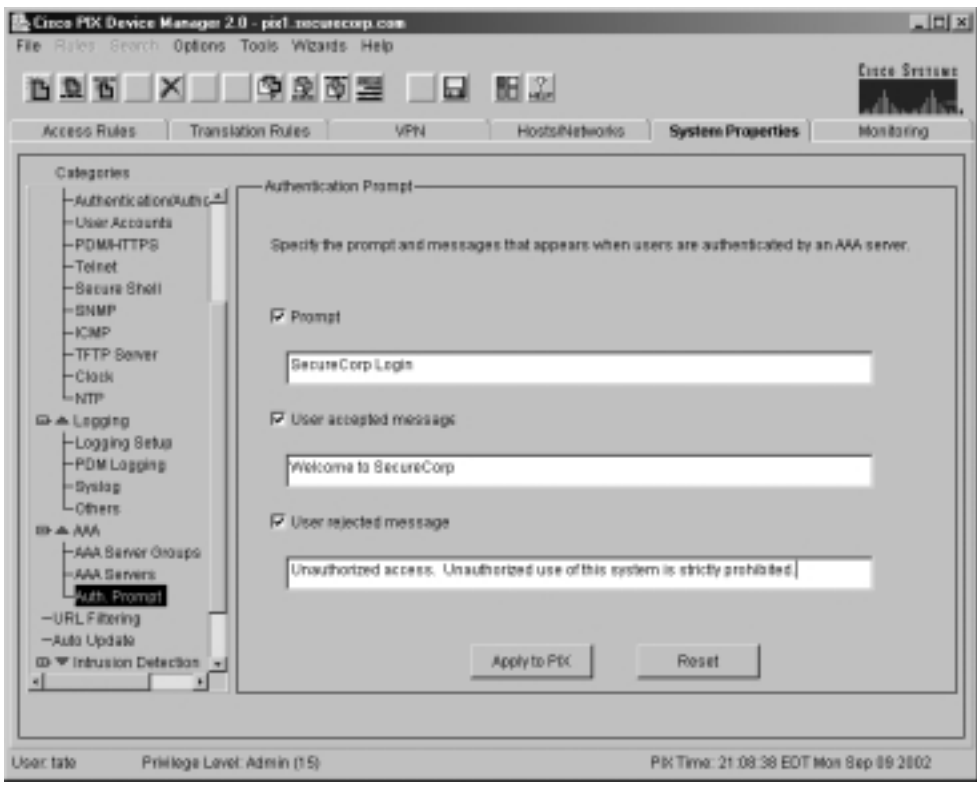
Figure 9.34 The Add AAA Server Window



To add a new server, select either **TACACS+** or **RADIUS** from the **Server Group** pull-down list. Next, select the interface to be used for AAA from the **Interface Name** pull-down list. Finally, type the new AAA server address in the **Server IP Address** field, select an authentication key in the **Key** field, and determine the AAA timeout period in the **Timeout** field.

The final AAA subcategory is Authentication Prompt and is shown in Figure 9.35.

Figure 9.35 AAA: The Authentication Prompt Screen



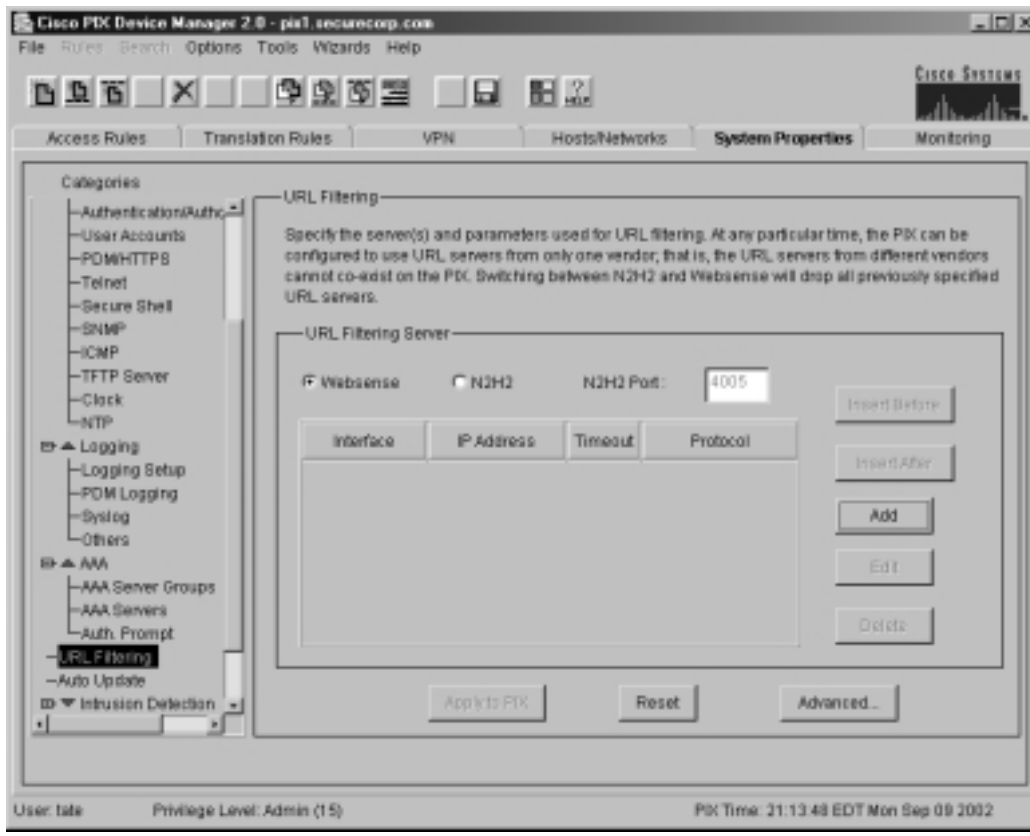
From the Authentication Prompt subcategory, you can customize the authentication prompts displayed to the user during the login process.

The URL Filtering Category

The ability to restrict user Web access to specific Web sites is desirable in some organizations for ethical and efficiency reasons. PDM provides a graphical interface to the PIX firewall URL filtering mechanisms via the URL Filtering

category available from the System Properties tab. The URL Filtering screen is shown in Figure 9.36.

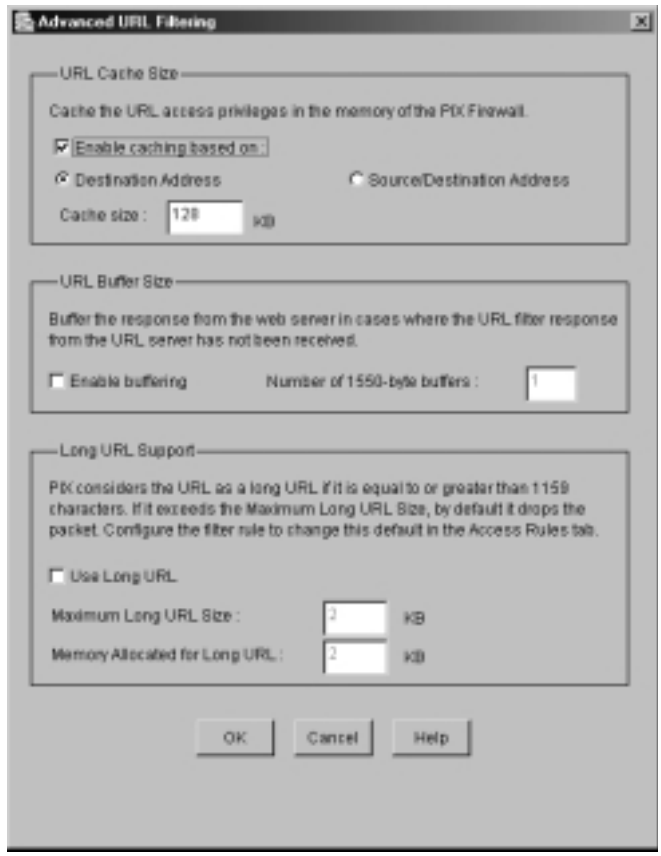
Figure 9.36 The URL Filtering Screen



As you can see, you can configure the PIX firewall to use third-party services to determine appropriate Web content. Currently available services include Websense and N2H2. To add a new server to the PIX configuration, select the type of server from the **URL Filtering Server** radio buttons. Then click the **Add** button and select the PIX interface used in filtering, IP address of the filtering server, and the protocol to be used for filtering. Other attributes are configurable from the Advanced URL Filtering window, as shown in Figure 9.37.

After configuring URL Filtering properties, you must add specific rules to the PIX firewall. These rules are determined from the Filtering Rules radio button under the Access Rules tab. We discuss the Filtering Rules radio button later in the chapter.

Figure 9.37 The Advanced URL Filtering Window



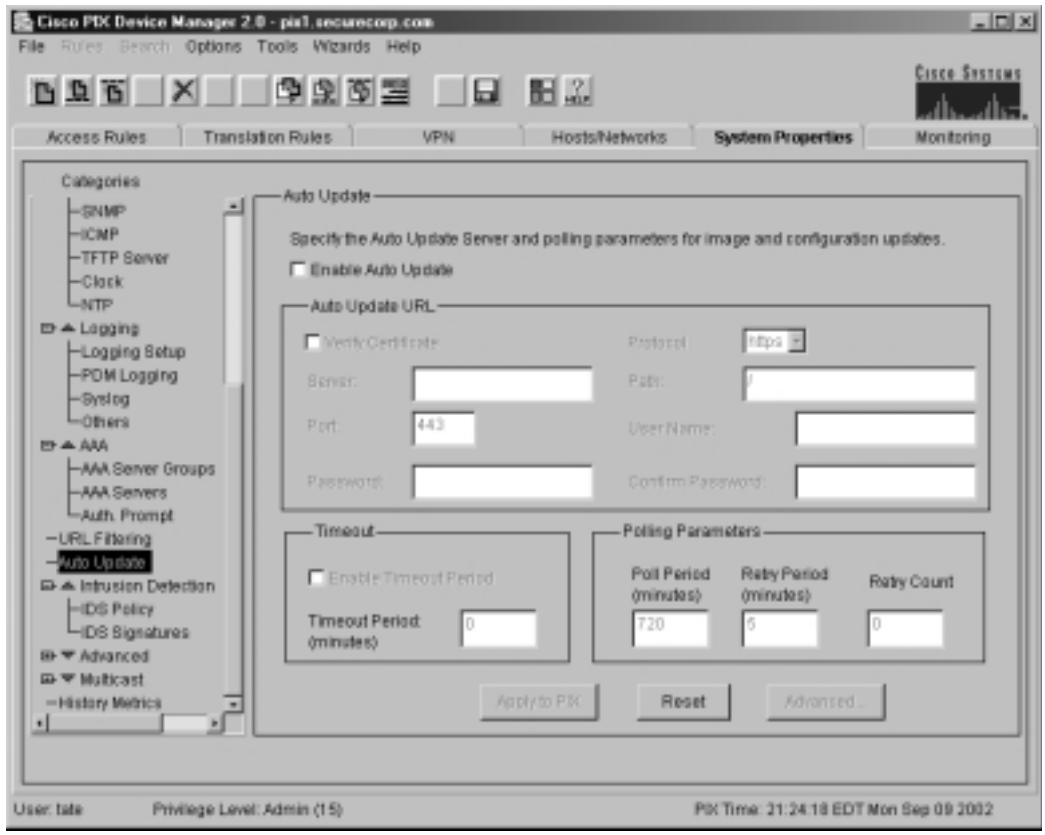
The Auto Update Category

PDM provides the ability to perform automated PIX configuration and image updates via HTTP or HTTPS connections. The automated update capability greatly simplifies firewall administration, especially in large corporate environments with multiple PIX firewalls. To configure automated updates, click the **Auto Update** category from the System Properties tab. To use the automatic update feature, you must first configure a Web server to store the configuration files and provide update services. The Auto Update screen appears, as shown in Figure 9.38.

To enable automatic updates, click the **Enable Auto Update** check box. Several attributes must be configured before auto-update will function properly. From the Auto Update URL section of the screen, determine the server address, port, password, and protocol. Additionally, you must specify the path to the

configuration file on the server. Other variables such as server timeout and polling parameters can also be configured from this screen.

Figure 9.38 The Auto Update Screen



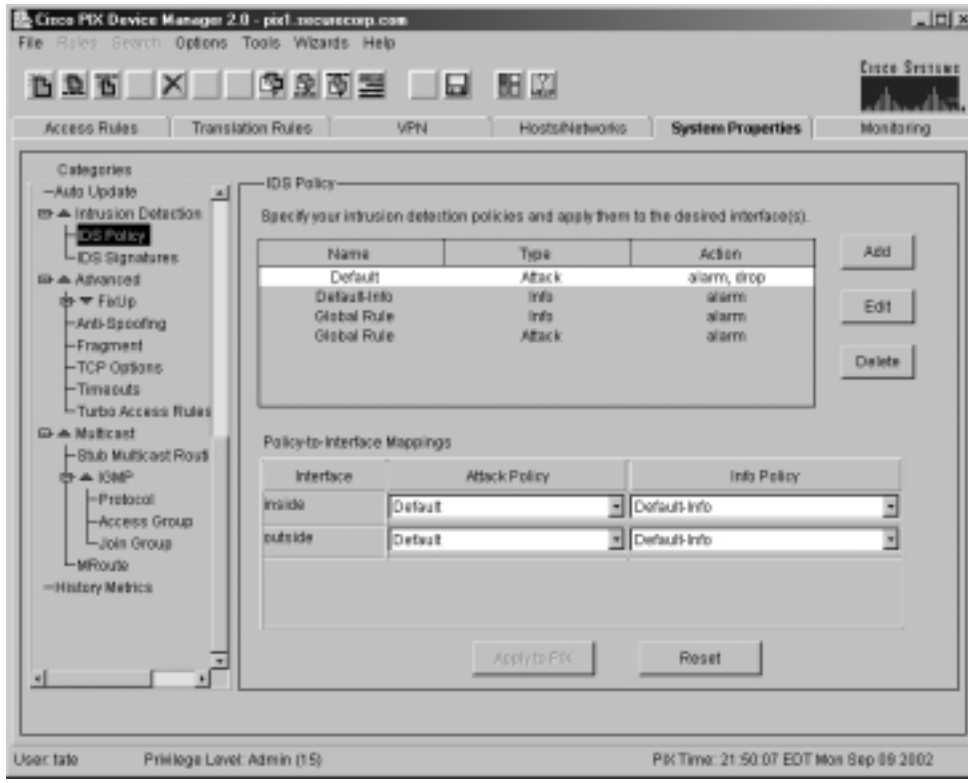
NOTE

Cisco Secure Policy Manager (CSPM) can be used as an auto-update server.

The Intrusion Detection Category

The PIX firewall includes intrusion detection capabilities that can be applied in various ways to the firewall interfaces. These capabilities can be controlled from the Intrusion Detection category. To enable IDS on the firewall, select the **IDS Policy** subcategory. The IDS Policy screen appears, as shown in Figure 9.39.

Figure 9.39 The IDS Policy Screen



To enable intrusion detection on the firewall, you must first create a policy and then apply that policy to an interface. Two types of policies are available: Attack and Info. To create a new policy, click the **Add** button from the IDS Policy screen.

From the Add IDS Policy window, determine a policy name and select the policy type by clicking the **Attack** or **Information** radio button. Finally, select an action to perform when the policy is triggered by clicking any of the **Drop**, **Alarm**, and **Reset** check boxes.

Once a policy has been created, it can be mapped to a specific PIX interface in the Policy-to-Interface Mappings section of the IDS Policy screen. To map a policy, select the specific policy from the pull-down list for each interface.

Administrators can also determine the type of intrusion signatures to detect on the PIX firewall. These signatures can be added and removed from the PIX firewall configuration by clicking the **IDS Signatures** subcategory. By default, all signatures are enabled on the PIX. To remove specific signatures, highlight the

signature you want to remove and click the **Disable** button to move the signature to the Disabled field.

The Advanced Category

The Advanced category permits the tuning of granular attributes available on the PIX firewall. These attributes include fixup capabilities, antispoofing, and TCP parameters. We discuss all these options in this section.

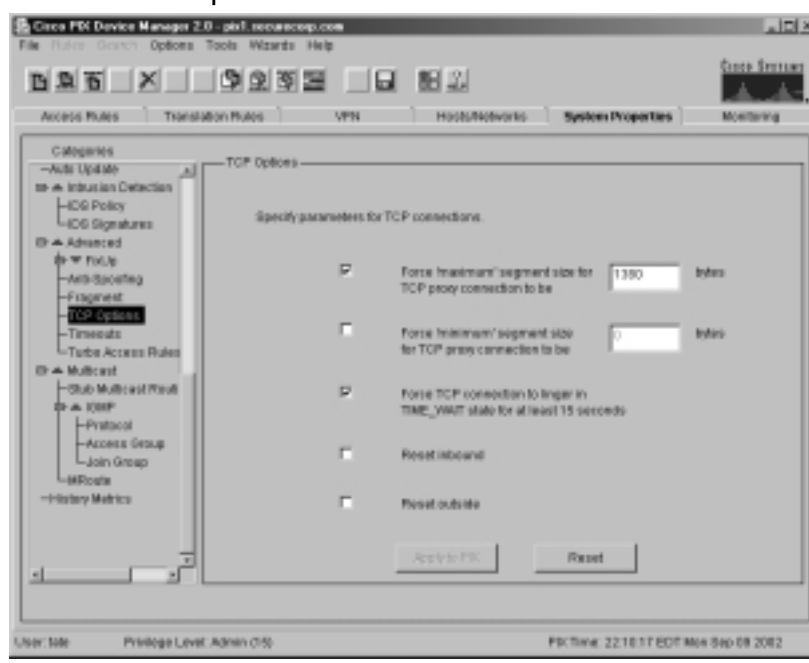
The fixup functionality available on the PIX firewall can be enabled and disabled from the Advanced category. Click the **FixUp** subcategory to view the current fixup configuration. To enable, disable, or customize any of the fixup protocols, click the specific subcategories.

Antispoofing, also known as *reverse-path forwarding (RPF),* is controllable from the Advanced category. From this subcategory, click the check boxes beside the specific interface to enable or disable antispoofing.

Fragment parameters can be established for each interface from the Fragment subcategory. These options include Size, Timeout, and Chain Length for each interface.

TCP options are also configurable from this category. Click the **TCP Options** subcategory to modify the options, as shown in the TCP Options screen (see Figure 9.40).

Figure 9.40 The TCP Options Screen



To enable a specific connection parameter, click the check box beside the variable and add attributes to the parameter as necessary.

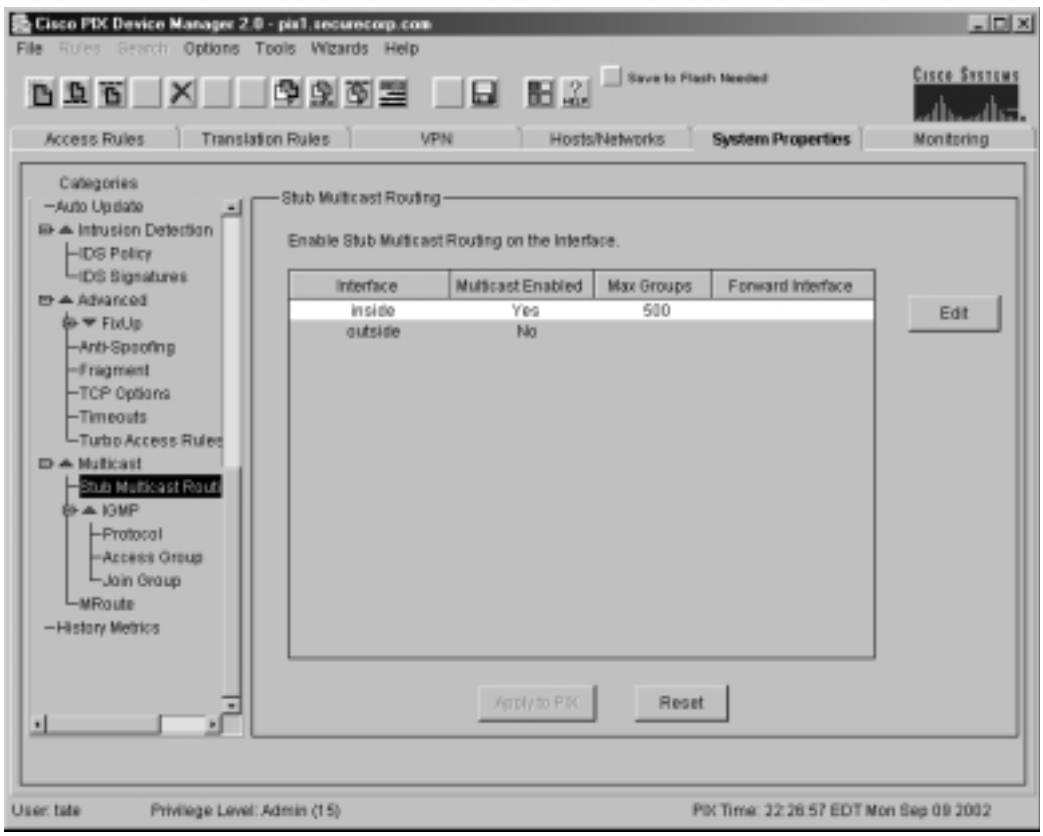
Timeouts can be determined from the Timeouts subcategory. Many timeout values are configurable on the PIX firewall. Some of these are connection, half-closed, and authorization timeout values.

If your firewall model supports it, you can configure Turbo ACLs from the Turbo Access Rules subcategory. Enable Turbo Access Rules by clicking the **Enable Turbo Access Rule Searches** check box from the Turbo Access Rules subcategory.

The Multicast Category

Multicasting can be enabled and disabled from PIX interfaces from the Stub Multicast Routing subcategory under the Multicast category, as shown in Figure 9.41.

Figure 9.41 The Stub Multicast Routing Screen



From this subcategory, you can also control the maximum number of multicast groups and whether IGMP forwarding is enabled on specific PIX interfaces.

IGMP parameters can also be controlled from the Multicast category. To determine the IGMP protocol, IGMP access groups, or join groups, click the specific subcategories under the IGMP subcategory.

Finally, multicast routes can be configured on the PIX firewall from the MRoute subcategory. To add a multicast route, click the **Add** button from the MRoute subcategory. The Add Multicast Route window is shown in Figure 9.42.

Figure 9.42 The Add Multicast Route Window

Configure the appropriate source and destination information, and click **OK** to add the new multicast route information to the PIX firewall.

The History Metrics Category

PDM collects metrics regarding many PIX firewall attributes by default. This capability can be controlled from the History Metrics category. To turn metric collection off, remove the check from the check box beside **PDM History Metrics**. PDM collects the following metric information on the PIX:

- Per-interface data such as:
  - Input and output bytes
  - Input and output packets
  - Input and output errors
  - Traffic

- Other metrics such as broadcasts, no buffer, giants, CRCs, overruns, underruns, collisions, late collisions, resets, deferred, lost carrier
- Available block count (4 bytes, 80 bytes, 256 bytes, and 1550 bytes)
- Free and used memory
- Perfmon information:
  - Xlates
  - Connections (UDP and TCP)
  - URL filtering/Websense
  - TCP Intercept
  - Protocol fixup
- IDS counters
- Failover statistics

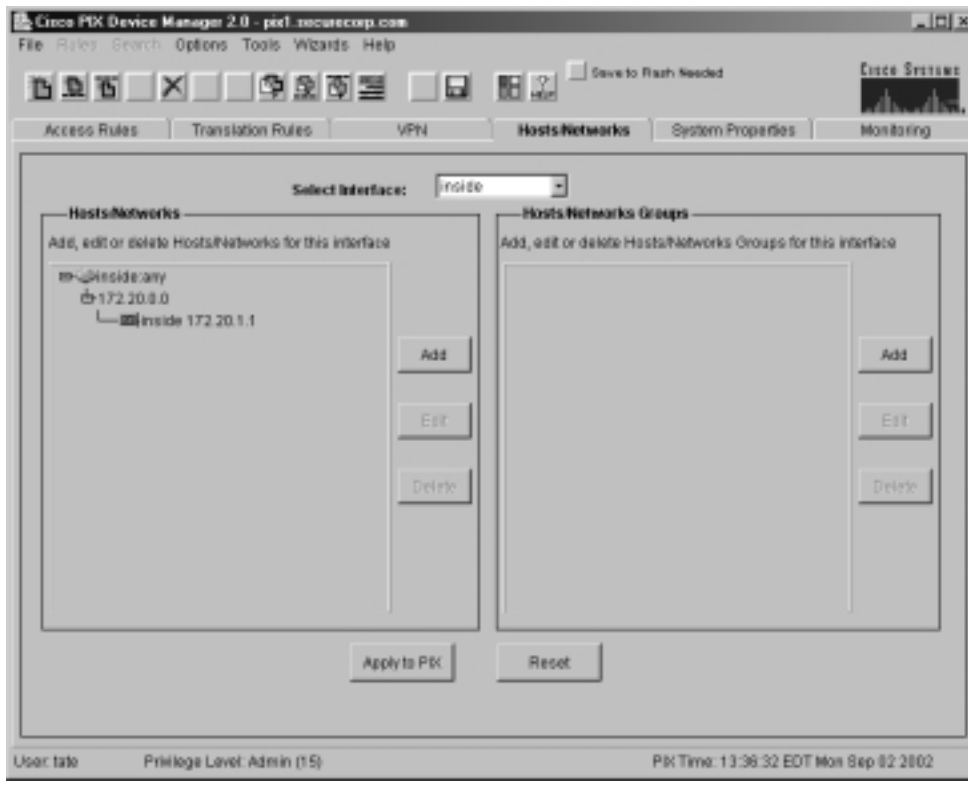
Maintaining Hosts and Networks

We have examined many of the properties configurable on the PIX firewall. At this point, you should have a good understanding of how to configure the PIX firewall itself. Now, let's look at how to configure the PIX firewall with regard to other objects on the network. Before adding access rules to permit or deny traffic through the firewall, you must configure host and network objects and/or groups from the Hosts/Networks tab.

From the Hosts/Networks tab, you can define specific attributes for remote and connected network and hosts such as IP information, NAT details, and routing configurations. These objects can represent internal resources such as mail servers and Web servers or external resources such as remote offices or networks. Click the **Hosts/Networks** tab to view the Hosts/Networks screen, shown in Figure 9.43.

This tab is organized into two sections: the Hosts/Networks section and the Hosts/Networks Group section. The Select Interface pull-down menu permits you to configure hosts and network objects available on specific PIX firewall interfaces. In the example described previously, the inside interface is configured with the network 172.20.0.0 and one specific, 172.20.1.1, which is the inside interface of the PIX firewall.

Figure 9.43 The Hosts/Networks Screen



Because we will be adding access rules to permit specific network traffic to internal servers, a host entry must be configured from this tab for each server. As an example, let's add a new Web server host to the internal network configuration so that we can add access rules later in the chapter. The host will have the attributes shown in Table 9.4.

Table 9.4 Web Server Host Attributes

Attribute	Value
Internal IP address	172.20.1.80
Mask	255.255.255.255
External IP address	192.168.1.20
Interface	Inside
Name	www

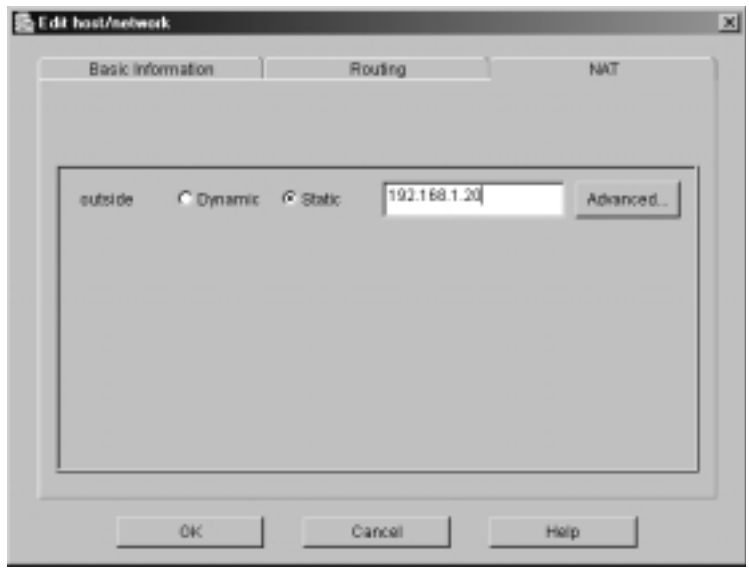
To add a new host, click **Add** from the Hosts/Networks section of the screen. The Create host/network basic information window appears. Fill in the appropriate fields (see Figure 9.44) and click **Next**.

Figure 9.44 The Create Host/Network Basic Information Window

Completing this form creates a new object in the PIX configuration. We use a 32-bit host mask in this example because we are adding a specific host. This mask should not be confused with the actual subnet mask on the host. By changing the mask in the **Mask** field using the drop-down menu, you could add a network object. After clicking **Next**, you will be prompted to configure NAT via the Create host/network NAT window, as shown in Figure 9.45.

From this window, you can configure either dynamic or static NAT, depending on the type of connectivity you want to allow to the new host. To permit only outbound connectivity (connectivity from a higher-security to a lower-security interface) from a host, select the **Dynamic** radio button. This choice dynamically translates the address of the added host to the specific NAT pool as determined by the Address Pool ID drop-down list. To permit both outbound and inbound connectivity (connectivity from a lower-security to a higher-security interface), click the **Static** radio button. This choice creates a one-to-one NAT mapping between the address of the added host and the address specified in the Static field.

Figure 9.45 The Create Host/Network NAT Window



In our example, we want to eventually permit inbound connectivity to our new internal Web server host. Therefore, click the **Static** radio button and add an externally available address such as **192.168.1.20**. This will configure the PIX firewall to translate our internal Web server's IP address of 172.20.1.80 to 192.168.1.20 and vice versa when traffic traverses the PIX firewall interfaces. Click **OK** to add the new host information to the PIX firewall configuration and return to the Hosts/Networks screen.

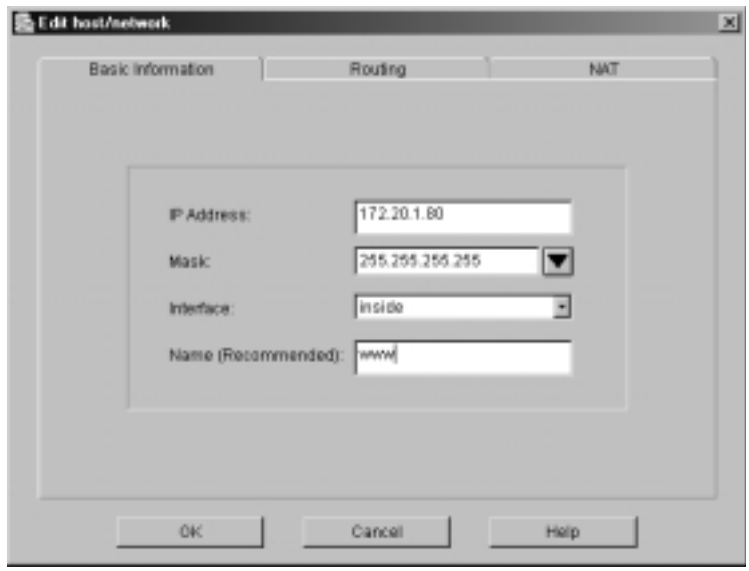
For practice, add a second example host object using the procedure we discussed. This time, however, click **Finish** after completing Create host/network basic information window. We will configure NAT for this host later in the chapter. Use the attributes listed in Table 9.5.

Table 9.5 Mail Server Host Attributes

Attribute	Value
Internal IP address	172.20.1.25
Mask	255.255.255.255
Interface	inside
Name	mail

From the Hosts/Networks tab screen, highlight the new Web server (www) object and click **Edit**. The Edit host/network window appears, as shown in Figure 9.46.

Figure 9.46 The Edit Host/Network Window

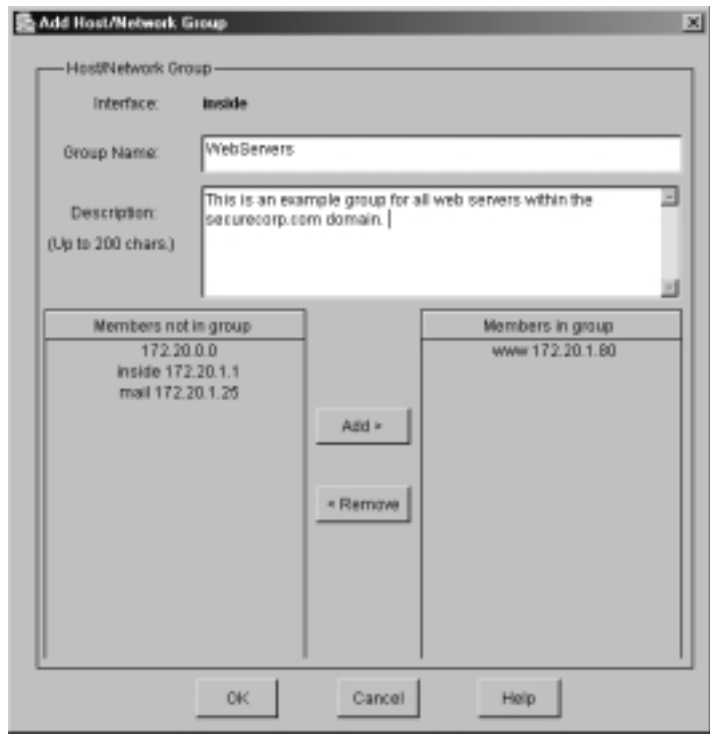


From this window, you can modify the host attributes added previously and add host or network specific routing information. For instance, if you add a network object to the PIX configuration and need to add a specific route statement for that network, you can do this from the Routing tab on the Edit host/network window. Alternatively, you can add routes via the System Properties tab Routing category, as previously described. Similarly, you can add or modify NAT information for specific hosts or networks from the NAT tab on the Edit host/network window or via the Translation Rules tab in the main PDM window. We discuss the PDM Translation Rules tab later in this chapter.

From the Hosts/Networks tab, you can also form groups of networks and hosts. This functionality simplifies rule management. Object grouping can also improve rule-processing efficiency on the PIX firewall. For example, if you have multiple servers that require HTTP connectivity, you could form a group object called *WebServers* and add all HTTP servers to the group, as shown in Figure 9.47.

To enable inbound access to the WebServers group, you simply add one access rule using the WebServers group instead of multiple, individual access rules for each Web server.

Figure 9.47 The Add Host/Network Group



Now that you understand how to add, modify, and delete host, network, and group objects, let's take a closer look at address translation.

Configuring Translation Rules

Address translation is widely used in networked environments to add additional layers of security and to conserve IP address space. To configure or view address translation using PDM, click the **Translation Rules** tab (see Figure 9.48).

From this screen, you can manipulate all configurations related to NAT, including translation rules, exemption rules, and global NAT pools. In our example, there is an existing static NAT rule, which pertains to the Web server host object we added previously. We can tell this is static translation by the icon in the Type column. The two NAT icons are shown in Figure 9.49.

From the Translation Rules screen, move the scroll bar at the bottom of the screen to the right until you can see the columns to the right of the DNS Rewrite column. Four Options columns should appear, as shown in Figure 9.50.

Figure 9.48 The Translation Rules Tab

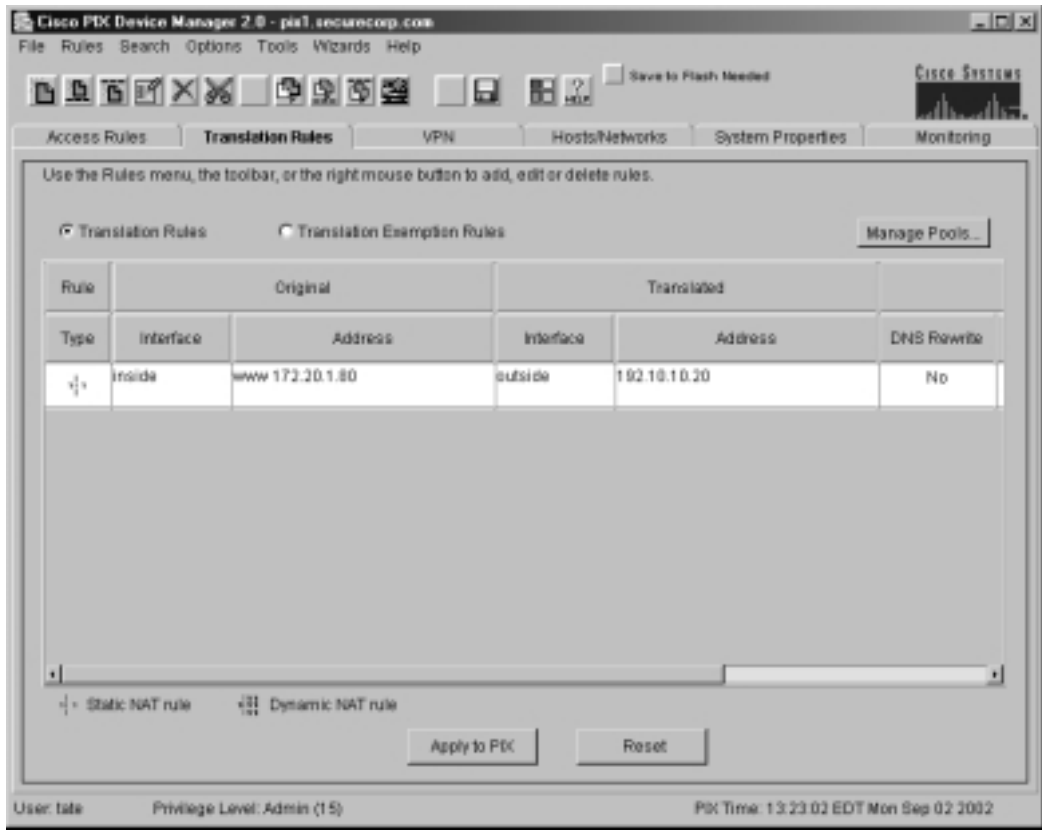


Figure 9.49 NAT Icons

Figure 9.50 NAT Options

Options			
DNS Rewrite	Maximum Connections	Embryonic Limit	Randomize Sequence Number

These options are available for use with all NAT rules. The DNS Rewrite option allows the PIX firewall to translate all DNS query responses through the firewall as specified in a NAT rule. With this functionality, administrators no longer need to maintain a split DNS architecture; the PIX firewall will translate

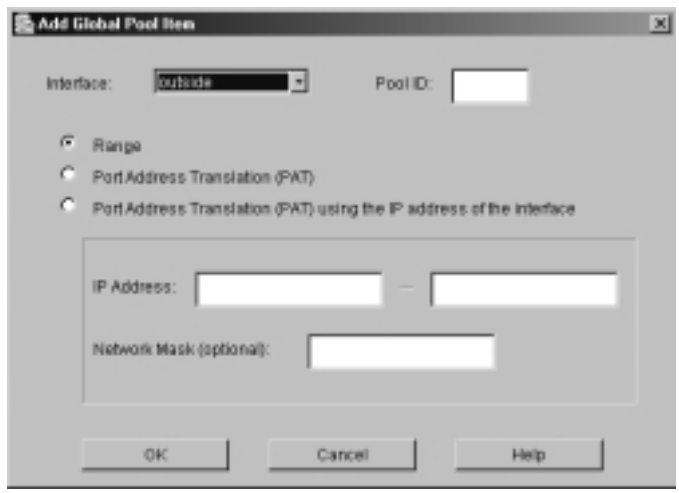
the responses from the internal DNS servers to external hosts. The remaining options relate to security and throttling mechanisms. We discuss these options in the following exercise.

You can also add rules to exempt specific entities from address translation. To do so, click the **Translation Exemption Rules** radio button and add a rule. This option is sometimes useful in situations with VPNs or when you do not want a specific server's address to be translated.

So far, we have added static NAT rules for the servers inside our network using the Hosts/Networks tab. Let's continue our example and add a dynamic translation rule for the remaining hosts inside our network. Doing so will provide outbound access for client workstations and other devices on our internal network while preventing inbound access to these devices.

First, create a global pool on which the dynamic translation will be based. Click the **Manage Pools** button to add a new address pool. The Manage Global Address Pools screen appears. Click **Add** to access the Add Global Pool Item window shown in Figure 9.51.

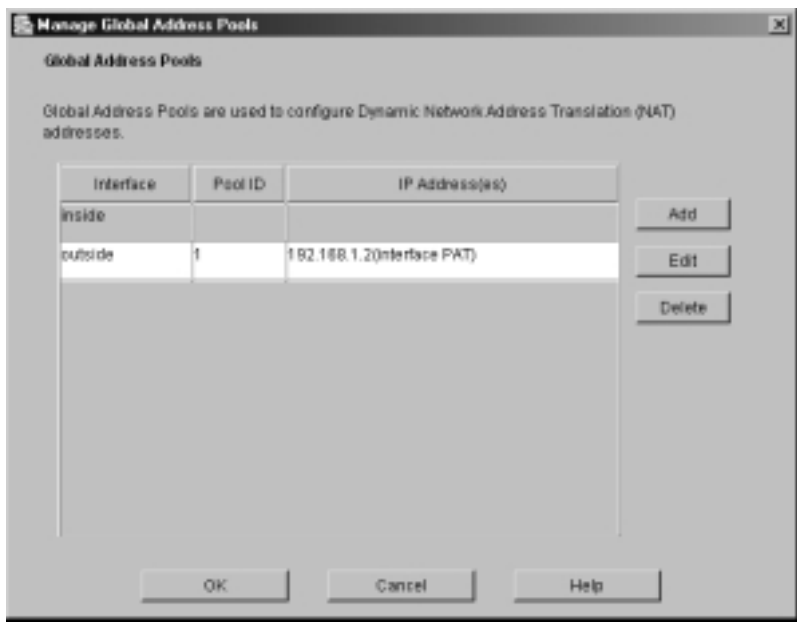
Figure 9.51 The Add Global Pool Item Window



From this window, you can create an outside or inside pool and define the pool ID. Furthermore, you can specify the type of translation to create such as a dynamic range, PAT, or interface PAT by clicking the **Range**, **Port Address Translation (PAT)** or **Port Address Translation (PAT) using the IP address of the interface** radio buttons, respectively. Based on your specific selection, you will also need to fill in the available fields before clicking **OK**.

For our exercise, we will configure interface PAT using the firewall's external interface. This method conserves IP address space on the external network. Alternatively, we could specify regular PAT and provide external IP address for the pool. To configure interface PAT, select the outside interface from the **Interface** pull-down menu and enter an integer such as one (1) in the **Pool ID** field. Do not use zero (0), because the pool ID of zero is reserved for no-NAT configurations. Click the third radio button, **Port Address Translation (PAT) using the IP address of the interface**, and click **OK**. No additional information is required because we have specified the external IP address of the PIX as the PAT address. The Manage Global Address Pools screen should appear, as shown in Figure 9.52.

Figure 9.52 Manage Global Address Pools



Note that the IP Address(es) column contains the external PIX firewall's IP address. Click **OK** to return to the Translation Rules tab.

This design allows the firewall's external IP address to be used in a dynamic NAT configuration. Next, from the **Rules** drop-down menu, select **Add** to create a new dynamic address translation on the firewall. The Add Address Translation Rule window appears, as shown in Figure 9.53.

Use the **Browse** button to display a list of available networks and hosts previously specified in the Hosts/Networks tab. Alternatively, you can type in the IP

address and subnet mask of the internal network (**IP Address: 172.20.0.0**, **Mask: 255.255.0.0**). Because we will be configuring PAT, click the **Dynamic** radio button and select **1** from the **Address Pool** drop-down list. This choice corresponds to the global pool ID we added in the previous step.

Figure 9.53 The Add Address Translation Rules Window

Click the **Advanced** button to view the Advanced NAT Options window. From this window, you can manipulate the options visible from the Translation Rules screen, such as DNS Rewrite. When finished, click the **OK** button. From the Translation Rules screen, click **Apply to PIX** to update the firewall and make the changes take effect. Now internal hosts should be able to access external resources.

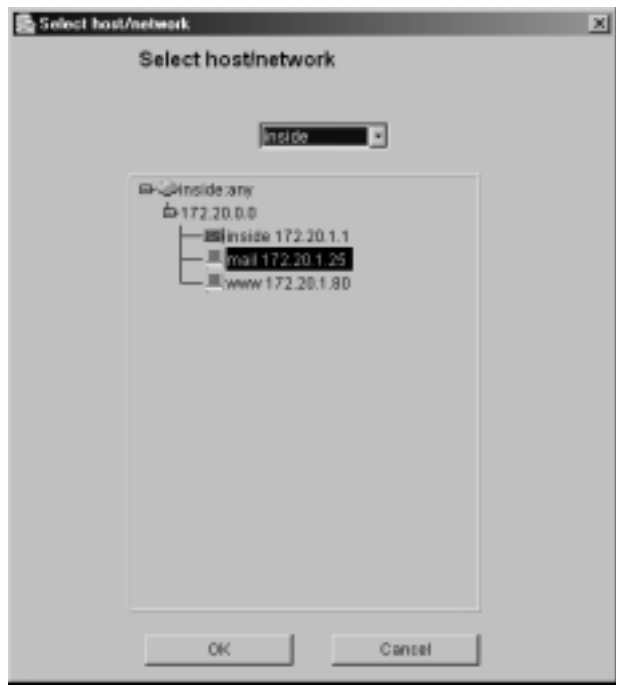
In SOHO environments where external IP space is limited, using interface PAT is extremely beneficial. For example, suppose you only have one static external IP address provided by your ISP. Your only option would be to use interface PAT for both inbound and outbound connections. Let's add a mail server using this premise.

NOTE

From the Add Address Translation Rule window, it is possible to specify all hosts by entering **0.0.0.0 0.0.0.0** in the **IP Address** and **Mask** fields. It is recommended that you specify each network to be translated, however, so that you have a full understanding of the networks traversing outbound through your firewall. This practice is extremely beneficial in large networks.

Assuming that you have already added a host object from the Hosts/Networks tab, now add a static translation rule. To do so, click **Add** from the **Rules** drop-down menu again. Click the **Browse** button and select the **mail** host object, as shown in Figure 9.54.

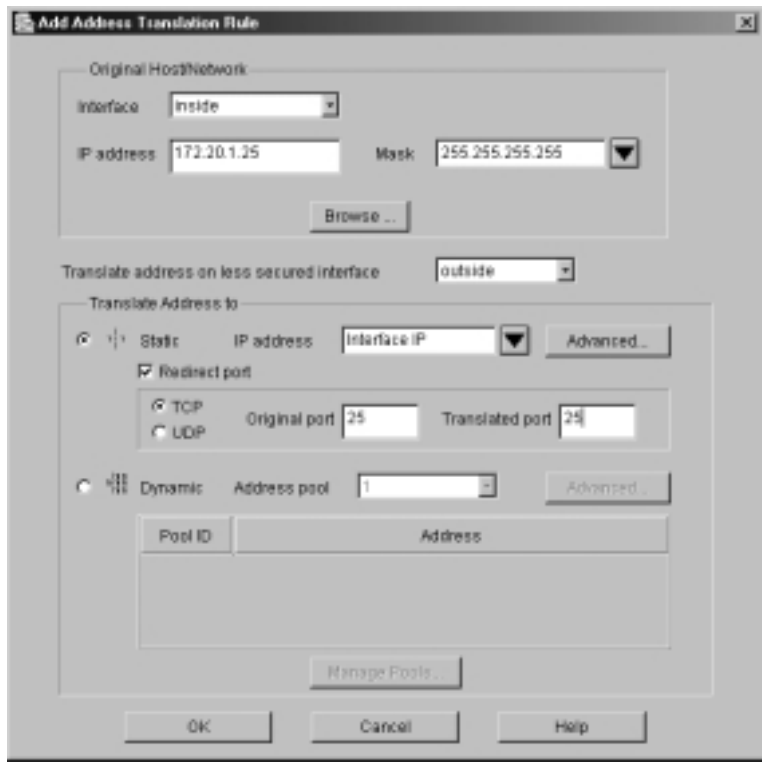
Figure 9.54 The Select Host/Network Window



Next, click the **Static** radio button and select **Interface IP** from the **IP address** pull-down list. Normally, this action would be sufficient to create a static NAT configuration similar to that already configured for the Web server. However, we already added an overall interface PAT rule for all internal networks. Therefore, we must specify actual ports to redirect as well. To do so, click

the **Redirect Port** check box and the **TCP** radio button. In the **Original Port** and **Translated Port** fields, type **25**, which is the TCP port for SMTP (mail) services. The Add Address Translation Rule window should appear, as shown in Figure 9.55.

Figure 9.55 Static PAT: The Add Address Translation Rule Window



Click **OK** to add the rule. You might be prompted with a warning message reminding you that the new configuration overlaps with the existing internal network interface NAT rule. Click **Proceed** to continue.

This configuration creates a static address translation mapping between the firewall's external IP address and the internal mail server IP address 172.20.1.25, but only for TCP port 25.

Next, let's add access rules to allow traffic through the firewall for these new servers.

Configuring Access Rules

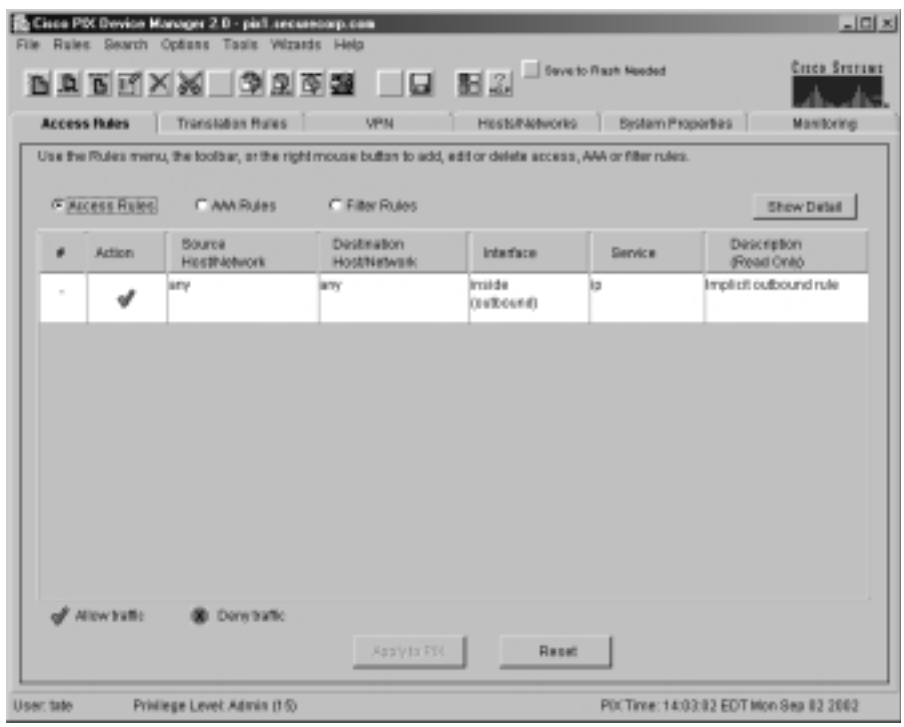
Once NAT has been successfully configured as shown in the previous exercise, internal clients should be able to access external resources. Even though a specific rule has not been manually added to allow such outbound access, it is implied through the configured interface security levels.

Using Cisco parlance, traffic is always permitted from firewall interfaces with higher security levels to interfaces with lower security levels. For instance, in the example network architecture, the external interface of the firewall at address 192.168.1.2 has a security level of 0, and the internal interface of the firewall at address 172.20.1.1 has a security level of 100. This allows internal traffic to traverse the firewall outbound without expressly permitting it.

However, this implied rule is reversed for traffic traversing from a lower security-level interface to a higher security level. Such traffic coming from outside networks to inside networks is always implicitly denied unless permitted. Therefore, you must add an access rule to permit any inbound traffic.

To add access rules, click the **Access Rules** tab from the PDM main window. The Access Rules tab screen appears, as shown in Figure 9.56.

Figure 9.56 The Access Rules Screen

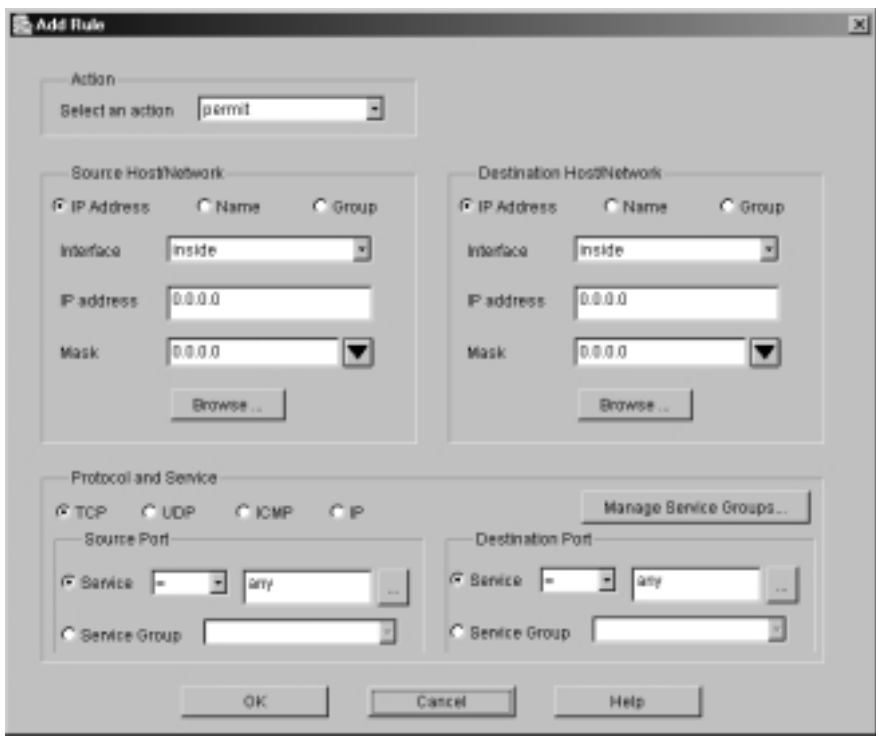


Note the existing rule automatically added by the PIX firewall implicitly permitting outbound access through the firewall. From this screen, you can configure access rules, AAA rules, and filter rules using the Access Rules, AAA Rules, and Filter Rules radio buttons, respectively. Access rules are used to permit and deny specific traffic through the firewall. AAA rules are used to configure AAA on specific connections permitted *through* the firewall. Finally, filter rules are used to permit or deny specific URLs or application functionality such as Java or ActiveX outbound. As with translation rules, you can manipulate all these rules via the PDM main menu drop-down menus, via the shortcut buttons, or by right-clicking your mouse in the rules screen.

Access Rules

For a detailed explanation of these rules, let's continue with our exercise, and permit Web and mail traffic to our example hosts. To permit access to the internal mail server, select **Add** from the **Rules** drop-down menu. The Add Rule window looks similar to Figure 9.57.

Figure 9.57 Add Access Rule



From the Add Rule window, there are four general areas you must configure. First, you must determine whether to permit or deny access with the rule. For this example, select **permit** from the **Select an action** pull-down menu. Next, you must specify source and destination information. Source and destination information can be in the form of IP addresses, names, or object groups. For this exercise, let's allow anyone to access our mail server. In the Source Host/Network section of the Add Rule window, click the **IP address** radio button. From the **Interface** pull-down menu, select **outside** and keep the IP address and Mask fields populated, as shown in Figure 9.57. Doing so specifies all possible networks arriving on the firewall's external interface.

Next, specify the mail server in the Destination Host/Network section of the Add Rule window. Click the **IP address** radio button and select **inside** from the **Interface** pull-down list. Click the **Browse** button and select the **mail** object from the popup window.

Now that we have determined the source and destination to permit access, let's configure the specific protocols and services to allow. Since this is a mail server, we should allow TCP port 25 (SMTP). Let's also permit TCP port 993 (Secure IMAP) so that our users can securely access their mail from remote locations. Previously, we would require two separate access rules to permit these two services. However, new functionality in the PIX firewall permits the formation of service group objects. This ability streamlines rule maintenance and facilitates more efficient rule processing. So, before adding protocols and services to our rule, let's configure a mail service group.

Click the **Manage Service Groups** button to access the Manage Service Groups window, as shown in Figure 9.58. Alternatively, you can access the Manage Service Groups window by selecting **Manage Groups** from the **Tools** menu of the main PDM screen.

From this window, you can create groups of TCP, UDP, and TCP-UDP services to be applied on access rules. Add a new TCP service group by clicking the **TCP** radio button and then the **Add** button. The Add Service window appears and is similar to the window shown in Figure 9.59.

From this window, specify a **Service Group Name** and add specific services to the group:

1. Type **MailServices** in the **Service Group Name** field; optionally, enter a description in the **Description** field. The PIX includes many common predefined services for use in service groups. From this list, scroll down, select **smtp,** and click the **Add** button.

Figure 9.58 The Manage Service Groups Window

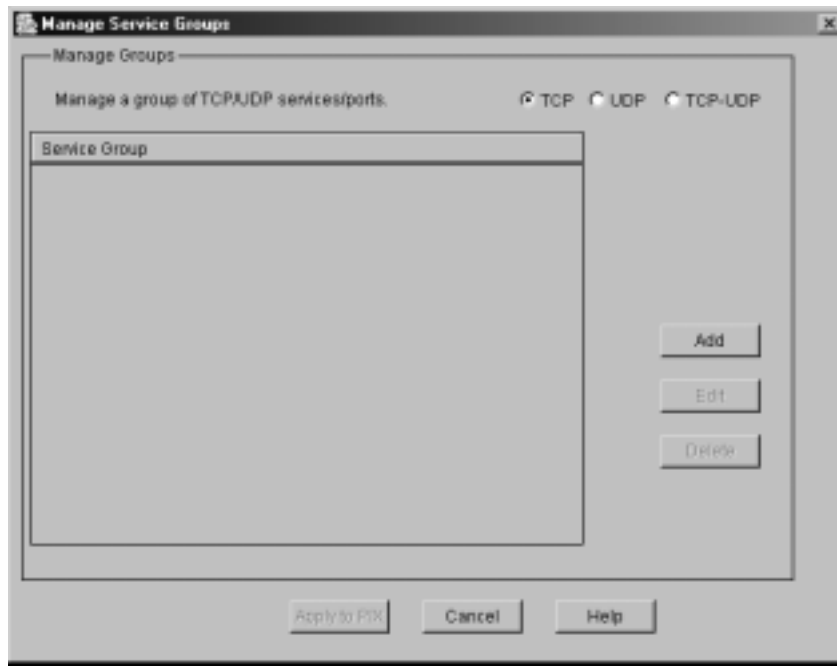
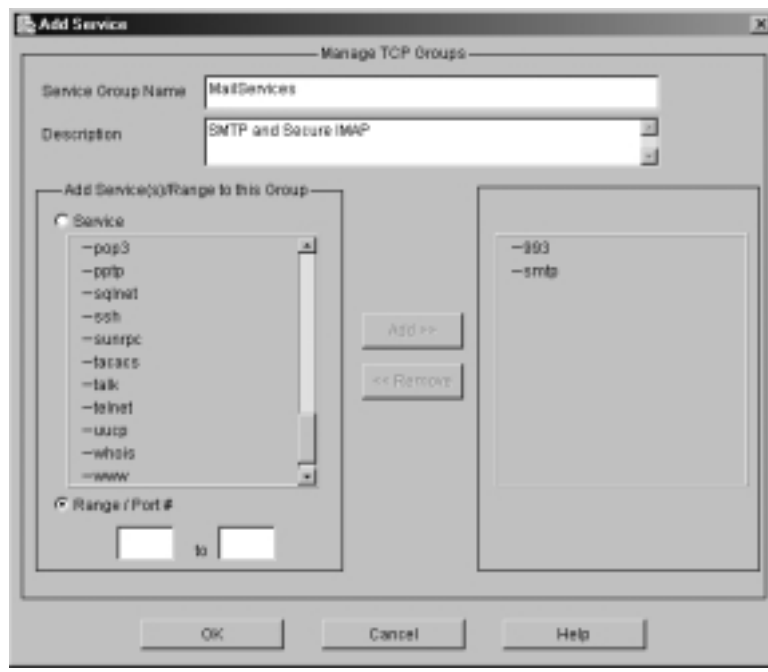


Figure 9.59 The Add Service Window



2. We need to add a custom service for secure IMAP because it is not predefined as a service. To do so, click the **Range/Port #** radio button and type **993** in the first field. Ranges of ports can also be created, but secure IMAP only requires TCP port 993.
3. Click the **Add** button to add the new service to the Services Group window on the left.
4. Click **OK** to add the group to return to the Manage Service Groups window.
5. From the Manage Service Groups window, click **Apply to PIX** and return to the Add Rule window.

Now that we have established a Service Group, let's add it to the mail server rule. In the Protocol and Service section of the Add Rule window, click the **TCP** radio button. Since the source port will be random, leave the Source Port section as is, with **Service = Any**. In the Destination Port section, click the **Service Group** radio button and select **MailServices** from the pull-down list.

NOTE

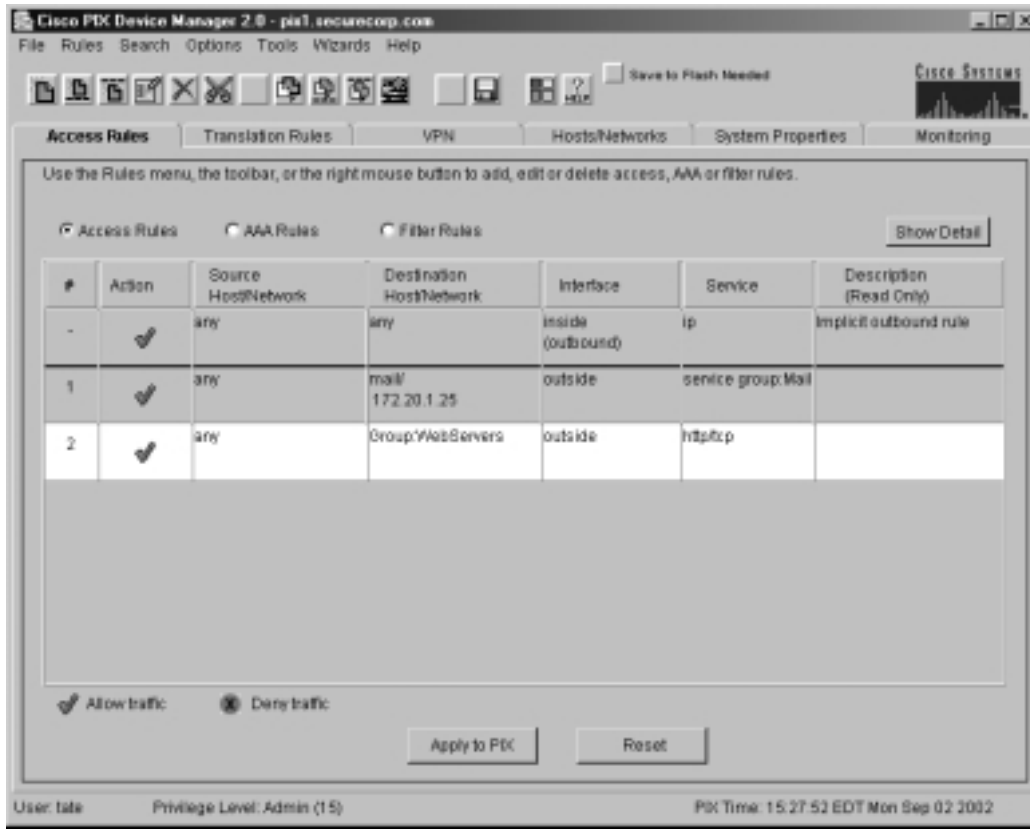
You might be required to refresh the PDM configuration before configuring a recently added Service Group in a rule.

Click **OK** to return to the Access Rules screen.

For practice, add a second access rule for the internal Web server. This time, instead of specifying an individual IP address from the Destination Host/Network section on the Add Rule window, click the **Group** radio button and select **WebServers** from the pull-down list. This choice designates any object included in the WebServers group we added in previous exercises and simplifies rule maintenance. In the Protocol and Service section of the Add Rule window, click the **Service** radio button and type **http** in the field. Alternatively, you can click the **…** button and select **http** from the services popup list. When finished, click **OK** to add the rule and return to the Access Rules window. The Access Rules tab window should now appear; it is similar to Figure 9.60.

After applying the new rules to the PIX firewall, mail and Web services should be permitted to your new servers through the firewall. Next, let's quickly look at the remaining rules screens, AAA Rules and Filter Rules.

Figure 9.60 The Access Rules Window



AAA Rules

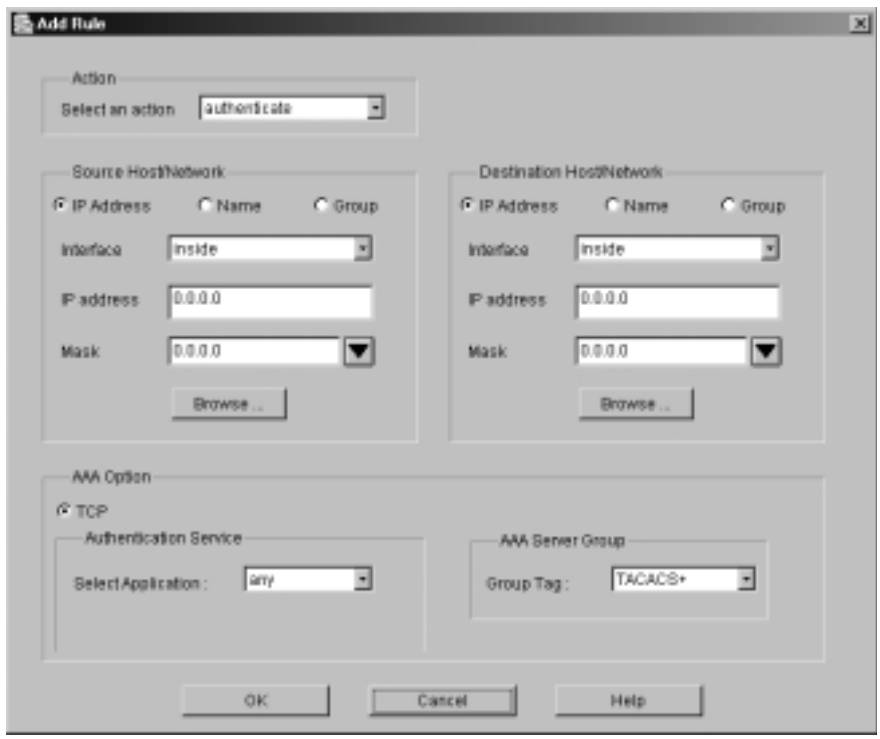
From the **Access Rules** tab, click the **AAA Rules** radio button to view the AAA Rules screen. From here, you can add rules regarding the operation of AAA servers and connectivity through the PIX firewall. For example, you could create a rule to authenticate, authorize, and audit Telnet connections through the firewall using a specific TACACS+ server.

Open the Add Rule window via the PDM Rules drop-down menu, the shortcut buttons, or by right-clicking your mouse in the rules screen. The Add Rule window appears (see Figure 9.61).

This window is similar to the previous Add Rule window. From here, you can choose various AAA actions, such as authenticate or account, based on source and destination variables. Furthermore, you can select specific application services such as Telnet or HTTP to be authenticated against a specific and previously

defined AAA server group using the Authentication Service and AAA Server Group areas of the Add Rule window, respectively.

Figure 9.61 The AAA Add Rule Window

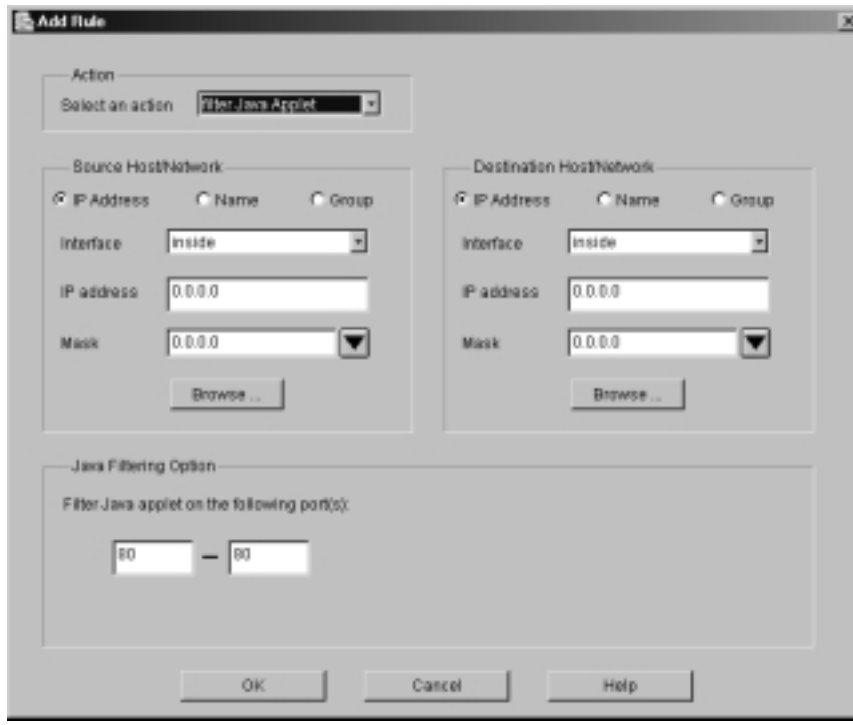


Filter Rules

The remaining rule-building functionality available on the Access Rules screen is Filter Rules. From the **Access Rules** tab, click the **Filter Rules** radio button. Filter rules allow you to permit or deny specific URLs to which users can navigate. This functionality can be provided for specific and individual URLs or based on interoperation with a URL filtering server as specified in the URL Filtering category in the System Properties tab. Valid URL filtering services are Websense and N2H2. Before configuring URL filtering, you must specify a URL filtering server from the System Properties tab.

From the Filter Rules screen, you can also configure the PIX firewall to permit or deny specific ActiveX or Java functionality. To do so, select **Add** from the **Rules** menu. The Add Rule window appears (see Figure 9.62).

Figure 9.62 The Add Filter Rule Window



Select **filter Java Applet** from the **Select an action** pull-down menu, then fill in the appropriate **Source Host/Network** and **Destination Host/Network** fields. Finally, specify the ports over which applets should be filtered. Typically, these values will be port 80 because that is the default service HTTP. When finished, click **OK** to return to the Access Rules tab.

At this point, you have configured the firewall itself, created specific hosts and network objects, created NAT rules, and permitted various inbound and outbound access through the firewall. Let's turn now to VPN configuration.

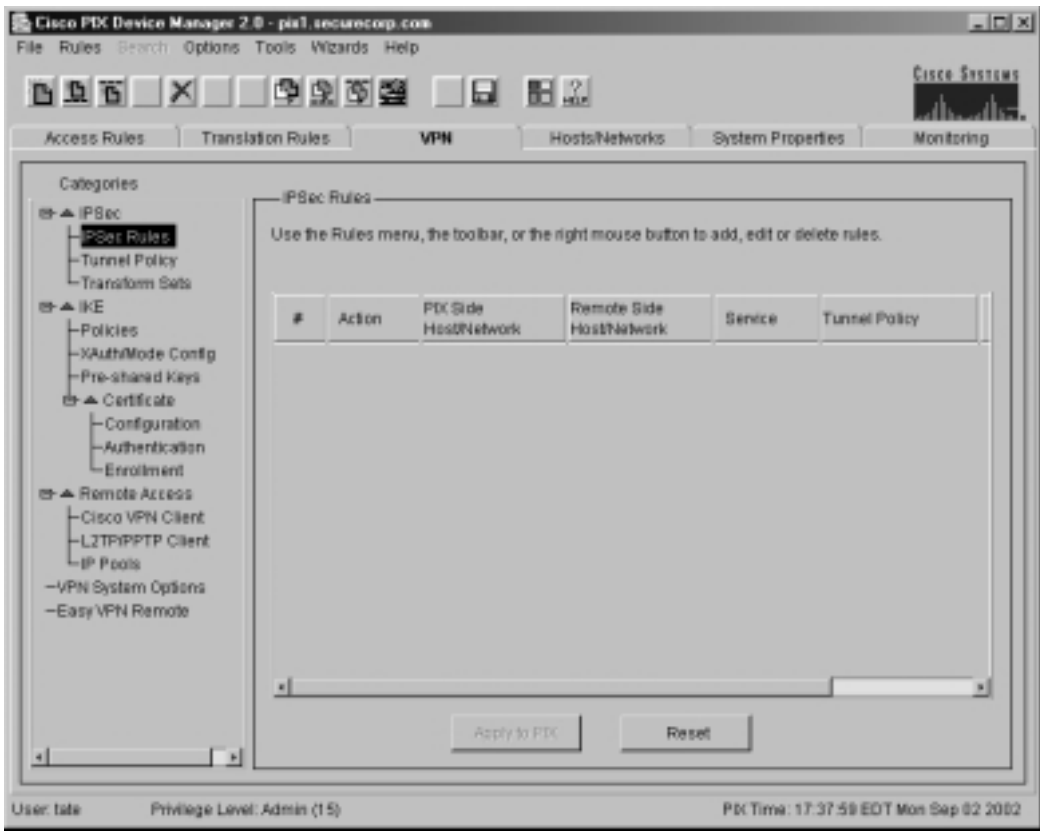
Configuring VPN

Virtual private networks are becoming increasingly prevalent in networks around the world. The application of VPNs within organizations has many benefits and will most likely continue to grow in popularity. PDM includes the capability to create, maintain, and monitor VPN access through the PIX firewall. In addition, a VPN Wizard is available for simplified VPN construction. This section covers the VPN-related capabilities of PDM and works through two exercises: configuring a site-to-site VPN and configuring VPNs for the Cisco software VPN client.

The PIX firewall is capable of supporting various tunneling protocols, including IPsec, PPTP, and L2TP. On the PIX, IPsec is used exclusively for site-to-site VPNs, whereas remote access or client VPNs can be built using any of the three protocols.

From the main PDM screen, click the **VPN** tab to access the VPN screen, as shown in Figure 9.63.

Figure 9.63 The VPN Tab



Similarly to the System Properties tab, the VPN screen includes categories on the left side of the screen that, when selected, alter the contents of the right side of the screen. The main categories include IPSec, IKE, Remote Access, VPN System Options, and Easy VPN Remote. Before discussing these categories and their specific subcategories, let's build a site-to-site IPsec VPN and an IPsec Cisco VPN Client VPN as an exercise using the VPN Wizard.

Configuring a Site-to-Site VPN

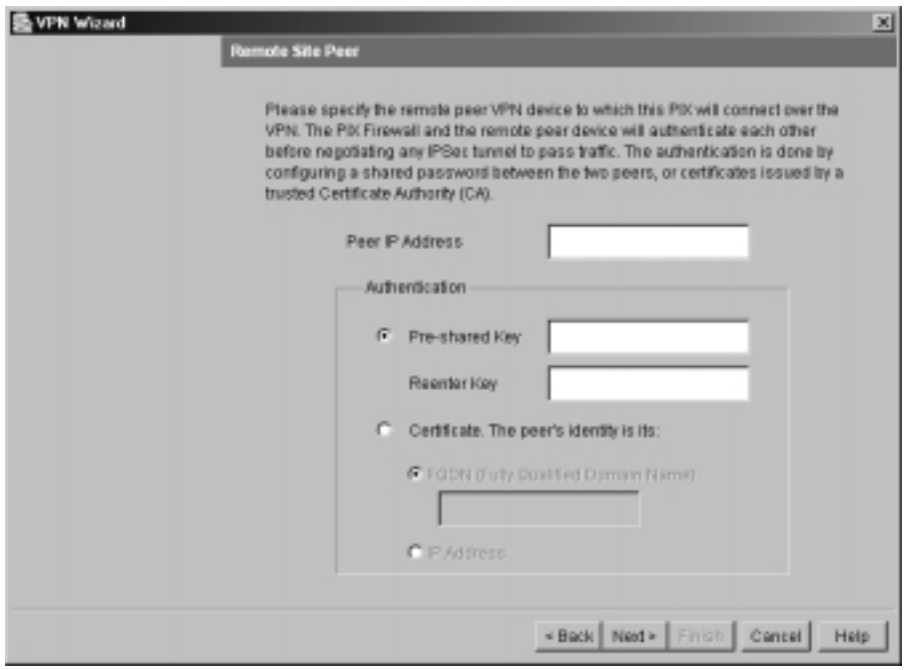
For our exercise, let's use our SecureCorp.com example network architecture to build a VPN between the Washington, D.C., PIX (PIX1) and the Prague PIX (PIX2). To build a site-to-site IPsec VPN using the VPN wizard, select **VPN Wizard** from the **Wizards** menu. The VPN Wizard window appears, as shown in Figure 9.64.

Figure 9.64 The VPN Wizard Window

Click the **Site to Site VPN** radio button and select **outside** from the **Select interface on which the VPN will be enabled** pull-down list. Click **Next** to proceed to the Remote Site Peer window, shown in Figure 9.65.

From this window, you can choose to use preshared keys or certificates. Using digital certificates is a more secure VPN tunnel configuration than shared keys. For simplicity, however, let's configure the site-to-site VPN using preshared keys.

Figure 9.65 The Remote Site Peer Window



Using our SecureCorp.com example architecture, type **192.168.2.2** in the **Peer IP Address** field. This is the external IP address of the PIX firewall named PIX2, located in Prague. Next, type an alphanumeric string in the **Pre-shared Key** and **Reenter Key** fields. This key string should be at least eight characters in length and should not be easily guessable. Remember the key entered in this step, because you will be required to enter it again when configuring the remote PIX firewall. After you click **Next**, the IKE Policy window appears, as shown in Figure 9.66.

Select appropriate **Encryption**, **Authentication**, and **DH Group** settings using the drop-down lists. It is important to remember the specific settings you select, because you will need to build an identical configuration on the remote PIX firewall.

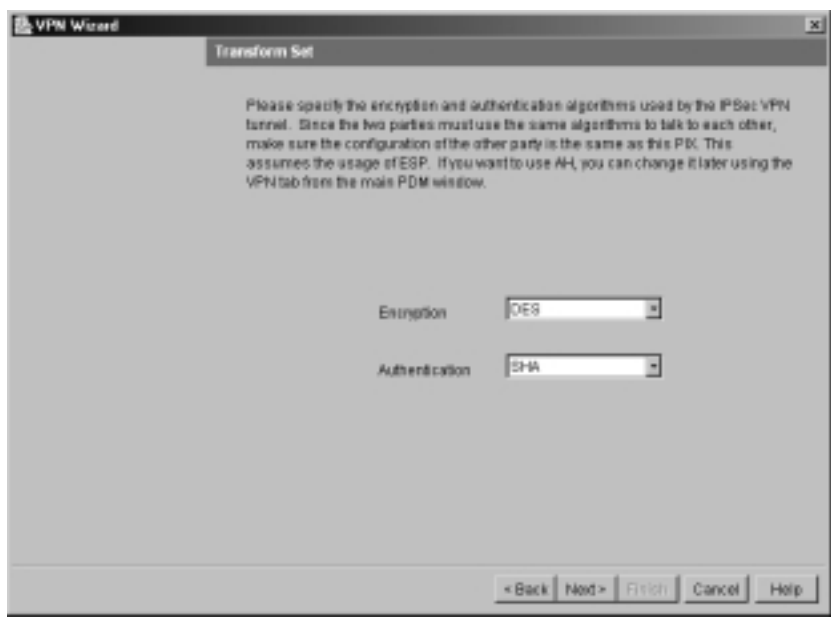
NOTE

3DES, which enables stronger encryption capabilities, is only available with a 3DES license from Cisco.

Figure 9.66 The IKE Policy Window

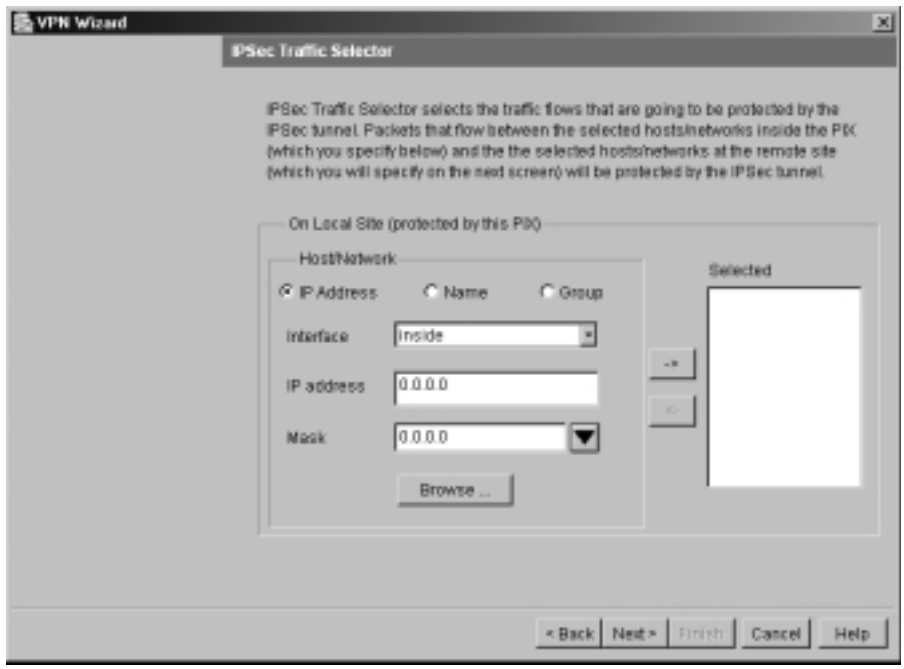
Click **Next** to proceed to the Transform Set window, shown in Figure 9.67.

Figure 9.67 The Transform Set Window



Similar to the IKE Policy window, the Transform Set screen permits you to select Encryption and Authentication variables. Again, remember your selections for configuration on the remote PIX firewall, and click **Next**. The next window you see is the IPSec Traffic Selector window, shown in Figure 9.68. From this window, you will determine the internal addresses that will traverse the tunnel.

Figure 9.68 The IPSec Traffic Selector Window



For the purposes of our exercise, we will use the entire internal network as the local site network. Alternatively, you could choose to only permit a subset of addresses across the VPN. Click the **Browse** button and select the internal network address, **172.20.0.0**. Click **OK** and, from the **IPSec Traffic Selector** window, click the **->** button. The address 172.20.0.0/16 should appear in the Selected window. Click **Next** to proceed.

Now that we have established the local site network to be transported across the VPN, we must select the remote network to which the VPN will connect. The next window to appear is quite similar to the one we just completed. From this window, enter the IP address of the Prague internal network, **172.16.0.0**, with a subnet mask of **255.255.0.0**. A popup window will appear, indicating that there is no host/network for 172.16.0.0 in the PIX configuration. When prompted, click **OK** to add the new network entry, and the Create Host/

Network window appears. Complete the necessary fields in the Create Host/Network window and click **OK**. Click the **->** button to add the new network to the Selected window.

Finally, click **Finish** to complete the VPN Wizard and return to the VPN tab. Before you can use the VPN, you must repeat the configuration process on the PIX firewall in Prague. This can be accomplished via a PDM session with the remote firewall, via the command line, or using other Cisco software such as CSPM. After finishing the remote firewall configuration, you are ready to begin testing and using the VPN.

Configuring for the Cisco Software VPN Client

You have built a static VPN connection between your global offices. Now let's enable IPsec-based remote access for traveling and telecommuting employees. For speed and simplicity, we will use the VPN Wizard again. After this exercise, we will consider manual VPN configuration; you will notice a dramatic difference between the two techniques in terms of ease of VPN creation.

From the PDM menu, click **Wizards** and select **VPN Wizard**. The VPN Wizard window appears. This time, select the **Remote Access VPN** radio button and click **Next**. You will be prompted to select a type of VPN from the many PIX remote access VPN capabilities, as shown in Figure 9.69.

Figure 9.69 The Remote Access Client Window

The PIX firewall supports IPsec tunneling from the Cisco software VPN client as well as Microsoft PPTP and L2TP protocols. Each type of VPN has inherent strengths and weaknesses. Each type of VPN has a VPN Wizard process unique to its requirements.

Since you are constructing a Cisco VPN client VPN, click the **Cisco VPN Client, Release 3.x or higher** radio button and click **Next**. The next wizard window is the VPN Client Group window, which allows you to create custom groups for shared remote VPN access. These groups use a preshared IKE key or certificates to connect and obtain group attributes. The VPN Client Group window is shown in Figure 9.70.

Figure 9.70 The VPN Client Group Window

From the VPN Client Group window, enter a group name in the **Group Name** field and establish a preshared key by clicking the **Pre-shared Key** radio button and typing a password for the group in the **Group Password** field. Alternatively, you can use certificates for authentication by clicking the **Certificate** radio button. Click **Next** to view the Extended Client Authentication window, as shown in Figure 9.71.

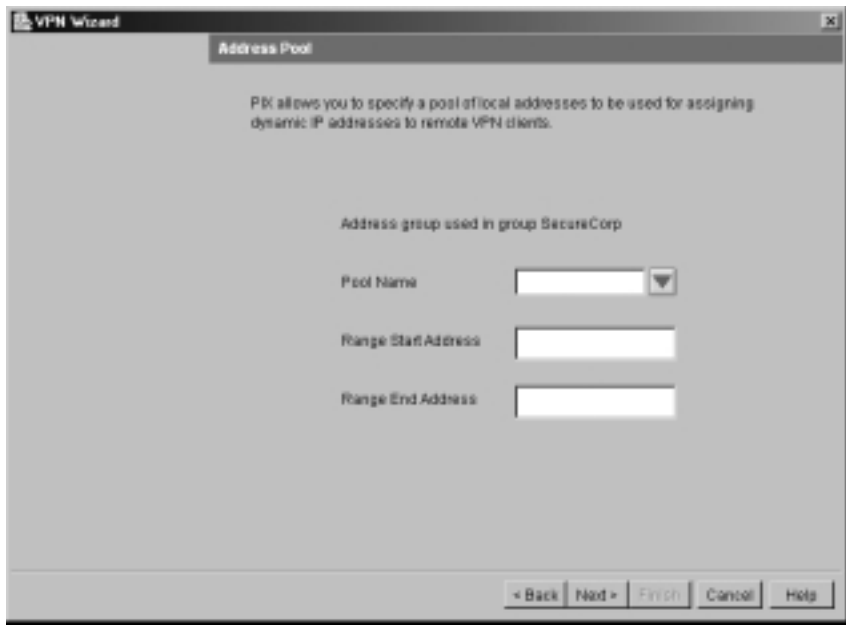
Figure 9.71 VPN Wizard: Extended Client Authentication Window

If you have an AAA server for authentication, click the **Enable Extended Client Authentication** check box and select a server group from the **AAA Server Group** pull-down list. This process configures the PIX firewall to consult the AAA server(s) in the specified server group for verification of user credentials as users request VPN access. From this window, you can also create a new AAA server group by clicking the **New** button. If your AAA server supports one-time passwords, click the check box beside **AAA server uses one-time password**.

For the purposes of this exercise, let's assume that we have no AAA server and will not use authentication for VPN connections. Therefore, uncheck the **Enable Extended Client Authentication** and click **Next**.

After specifying authentication variables as shown previously, you will be prompted to select or create a VPN client address pool from the Address Pool window, as shown in Figure 9.72.

Figure 9.72 The Address Pool Window

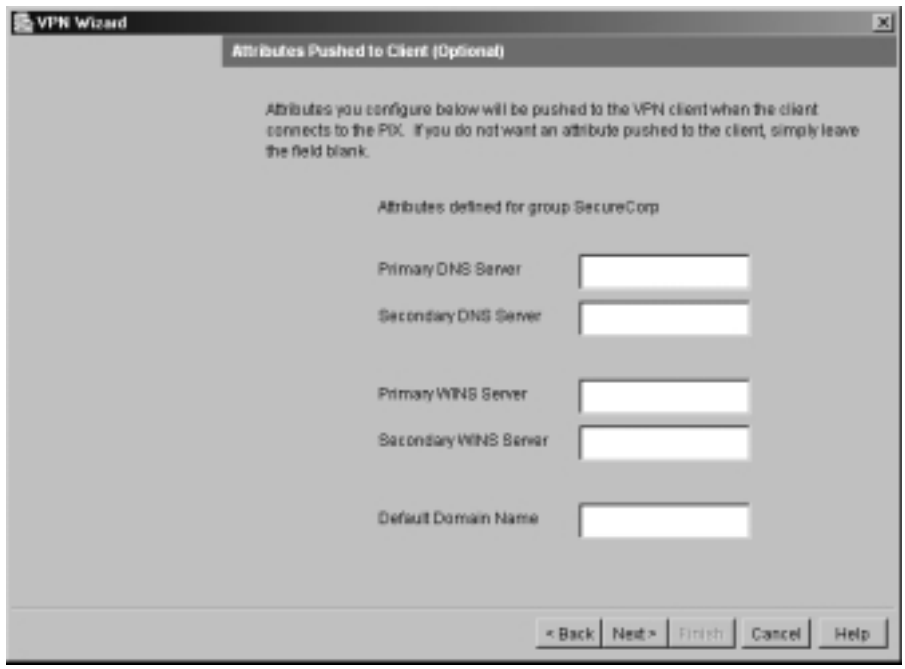


When clients connect via VPN, they are given an IP address to be used over the tunnel for the duration of the connection. These addresses are established from the Address Pool window. If you have already manually established an address pool, simply select the pool from the **Pool Name** pull-down menu. If you have not established an address pool, create a pool called *SecureCorpPool*. To create this pool, type **SecureCorpPool** in the **Pool Name** field and create an IP address range for the VPN clients in the **Range Start Address** and **Range End Address** fields. Use **172.20.200.0** as the Range Start Address and **172.20.200.30** as the Range End Address. Be careful not to create an address pool that conflicts with one already in use or that is being offered via an internal DHCP server. When finished, click **Next** to proceed.

The screen shown in Figure 9.73, the Client Attributes window, is where you can specify optional attributes to send to the VPN client upon connection. From the Client Attributes window, you can specify DNS and WINS servers as well as the default domain name.

In our example, use **172.20.1.53** and **172.20.2.53** as the Primary DNS Server and Secondary DNS Server, respectively. Leave the WINS Server fields blank, but type **vpn.securecorp.com** in the **Default Domain Name** field and click **Next**.

Figure 9.73 The Client Attributes Window



The next two wizard windows are the IKE Policy and the Transform Set windows, which are identical to the windows displayed in the site-to-site VPN Wizard. Like the site-to-site VPN, these windows establish some of the crypto parameters required for VPN setup. Several options will function with most VPN configurations, but it is important that the VPN client and server be configured identically. Choose the default options on these screens and click **Next** until you reach the NAT Exemption window shown in Figure 9.74.

In most instances, VPN clients allowed to the firewall are connecting for internal services. Therefore, it might be beneficial to permit VPN clients access to the actual IP address of internal servers without NAT application. To do so, you must configure specific networks (or all networks) to be exempt from NAT with regard to VPN clients. Additionally, you can configure split tunneling from this screen. Split tunneling allows VPN clients access to internal resources when necessary yet permits the client direct access to external resources when applicable. This configuration is advantageous because it conserves corporate bandwidth; clients are not required to route all traffic to the internal network for external resources. In some instances, administrators might want to disable split tunneling to increase security and better track VPN client network activity.

Figure 9.74 The NAT Exception Window

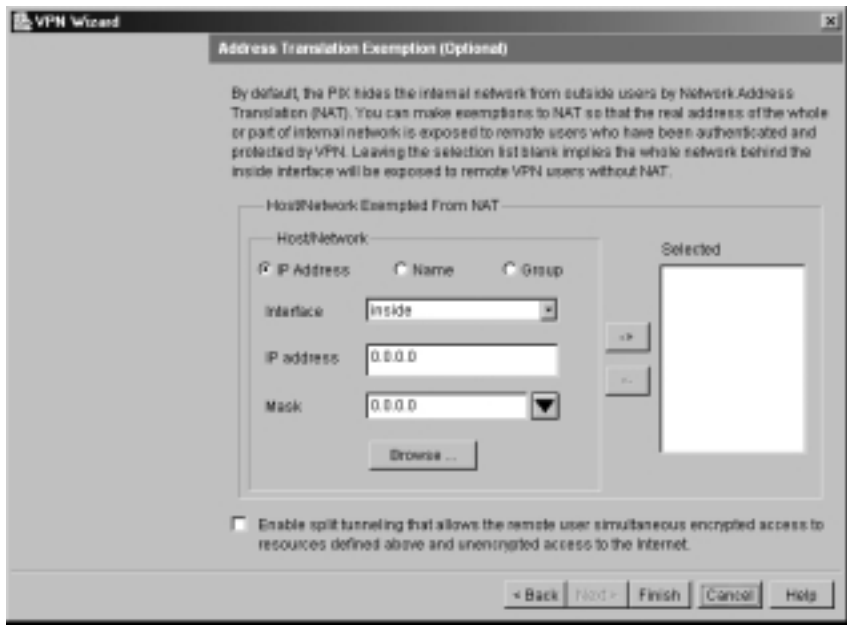


Figure 9.75 The Preview CLI Commands Window

In our example, let's make the internal network exempt from NAT and permit split tunneling. To do so, click the **Browse** button and select the internal network address of **172.20.0.0**. Then, click the **->** button to move the network

into the Selected field. Finally, click the check box to enable split tunneling, and click the **Finish** button to return to the VPN tab.

In eight easy steps, you have created remote VPN access to external clients. If you have preview commands enabled within PDM, you can see the relative simplicity of the VPN Wizard compared with manually creating a VPN via the CLI. Remember, you can configure preview commands from the Options | Preferences PDM main menu. Figure 9.75 shows the CLI commands the PIX configured for you in our example.

Now that you have configured site-to-site and Cisco software VPN client VPNs with the VPN Wizard, let's return to the VPN tab to discuss more specific categories.

From the VPN tab, you can now see the two VPN configurations present by clicking the **IPSec Rules** subcategory under the IPSec category. Note the difference in the two rules as created by the VPN Wizard. From here, you can add, modify, and delete IPsec rules using the Rules main menu bar, the shortcut buttons, or by right-clicking in the rules screen. Under the IPSec category are two other subcategories called Tunnel Policy and Transform Sets. From these subcategories, you can configure new and more granular policies, such as determining the Security Association Lifetime in terms of bytes or seconds. From the Tunnel Policy subcategory, you may also configure Perfect Forwarding Secrecy. The Transform Sets subcategory allows you to create new encryption and authentication groups as well as determine whether a VPN exists in transport or tunnel mode.

The second category available from the VPN tab is IKE. From this category, you can configure SA and IKE management policies. The Policies subcategory is shown in Figure 9.76.

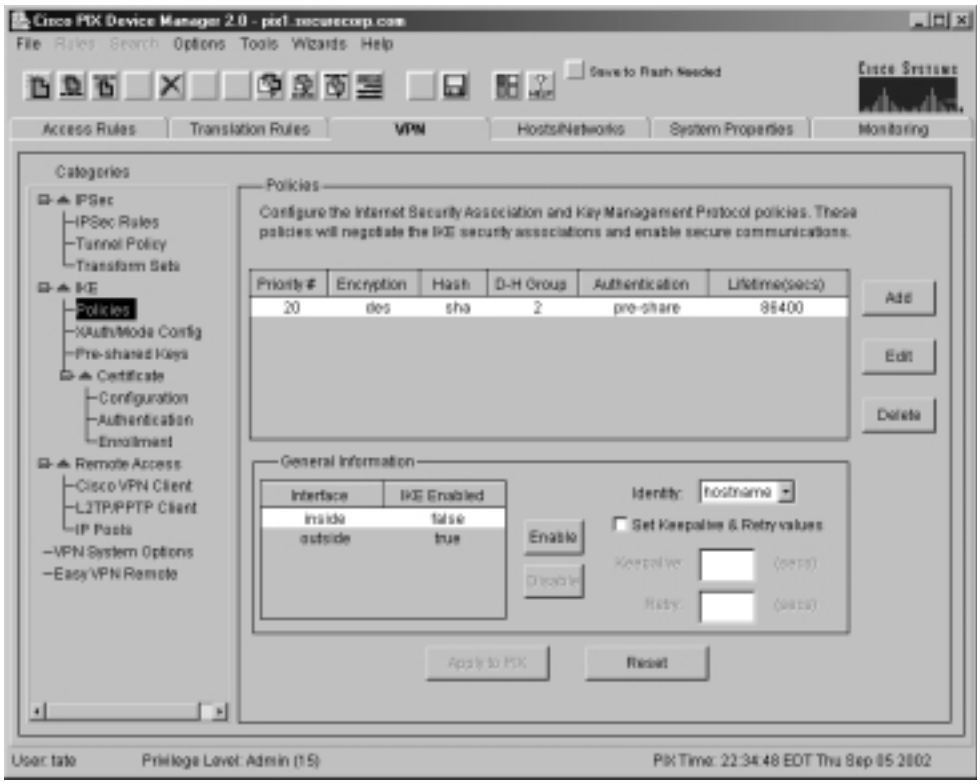
The IKE category also facilitates advanced configuration of authentication and preshared key information. A great deal of advanced certificate management is available from the Certificate subcategory. For instance, from the Certificate subcategory, you can generate requests to a certificate authority and manage existing certificates on the PIX firewall.

A third category on the VPN tab is Remote Access. From this category, you can add, modify, and delete the various remote access VPNs supported on the PIX firewall, such as Cisco VPN client, L2TP, and PPTP VPNs. From the Remote Access category, you can also configure IP pools for use with remote clients. All the functions and features from these and nearly all other VPN tab categories are available via the VPN Wizard through an intuitive interface.

The final two categories on the VPN tab are VPN System Options and Easy VPN Remote. From the VPN System Options category, you can determine whether the various VPN protocols are permitted to bypass security to establish

connections to the PIX firewall. This permits VPN connections without specific permit rule statements in the PIX firewall rule sets and is enabled by default when you use the VPN Wizard to build VPN configurations.

Figure 9.76 IKE: The Policies Screen



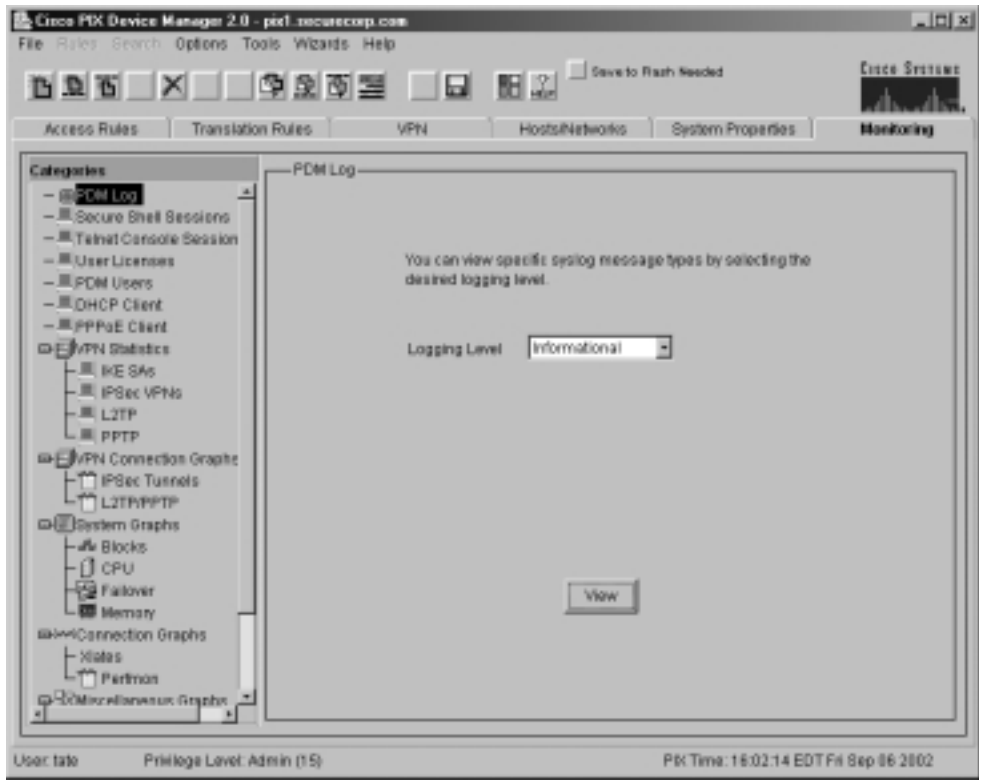
From the Easy VPN Remote category, you can configure the PIX firewall as an IPsec client to another PIX firewall, Cisco VPN Concentrator, or IOS device.

Monitoring the PIX Firewall Using PDM

One of the many beneficial aspects of PDM is the ability to visually monitor virtually every attribute of the firewall in near real time. Information such as VPN connectivity, resource utilization, history bandwidth trending, and administrative maintenance data can all be viewed via tabular or graphic reports with PDM. This functionality is available from the Monitoring tab of the PDM interface. Click the **Monitoring** tab from the main PDM interface to view the Monitoring screen shown in Figure 9.77.

Much like the System Properties screen, the Monitoring screen includes various categories on the left side of the screen. When you click a category, information pertinent to the category is displayed on the right side of the screen.

Figure 9.77 The Monitoring Tab



In the example in Figure 9.77, the PDM Log category is displayed. Clicking the **PDM Log** category allows you to view the current PDM log based on the logging level you select from the **Logging Level** pull-down list. After you click the **View** button, the log appears. From the PDM Log Viewer window, you can choose to clear, refresh, or close the window. To view the PDM log, you must enable PDM logging from the Logging category found on the System Properties tab.

Many of the categories, such as PDM Users or Telnet Console Sessions, describe sessions or statistical data in real time on the PIX firewall when you click the category. Others, such as VPN Statistics, VPN Connection Graphs, and System Graphs, show near real-time updated graphs and/or tables. We discuss all of these and more in this section.

Sessions and Statistics

The next several categories under the PDM Log category show session and statistical information related to connections and functionality on the PIX firewall. These categories include the following:

- Secure Shell Sessions
- Telnet Console Sessions
- User Licenses
- PDM Users
- DHCP Client
- PPPoE Client
- VPN Statistics

These categories are slightly different from those discussed later in this section, because they do not automatically update and they simply show numeric data rather than graphical output.

From the Monitoring tab, you can see information regarding the current administrative connectivity to the PIX. Furthermore, you can actively disconnect administrative users using PDM. For instance, the categories Secure Shell Sessions, Telnet Console Sessions, and PDM Users all display administrative connection information. The screens available for each of these categories are quite similar. Let's look at the Telnet Console Sessions and PDM Users categories as an example.

Click the **Telnet Console Sessions** category to view the Telnet Console Sessions screen. An example is shown in Figure 9.78.

In this example, one Telnet session is currently connected to the PIX firewall. The client name is SecureCorp-CSPM and the allocated virtual console is 0. The server name appears as a name rather than an IP address in this instance because the host SecureCorp-CSPM has been configured via the Hosts/Networks tab. To search for specific IP addresses connected to the PIX firewall via Telnet, type the IP address in the **Show sessions for this IP Address** field, and click the **Refresh** button. The Secure Shell Sessions screen is quite similar in nature to the Telnet Console Sessions screen.

The PDM Users category displays the currently connected PDM sessions. Click the **PDM Users** category to reveal the PDM Users screen shown in Figure 9.79.

Figure 9.78 The Telnet Console Sessions Screen

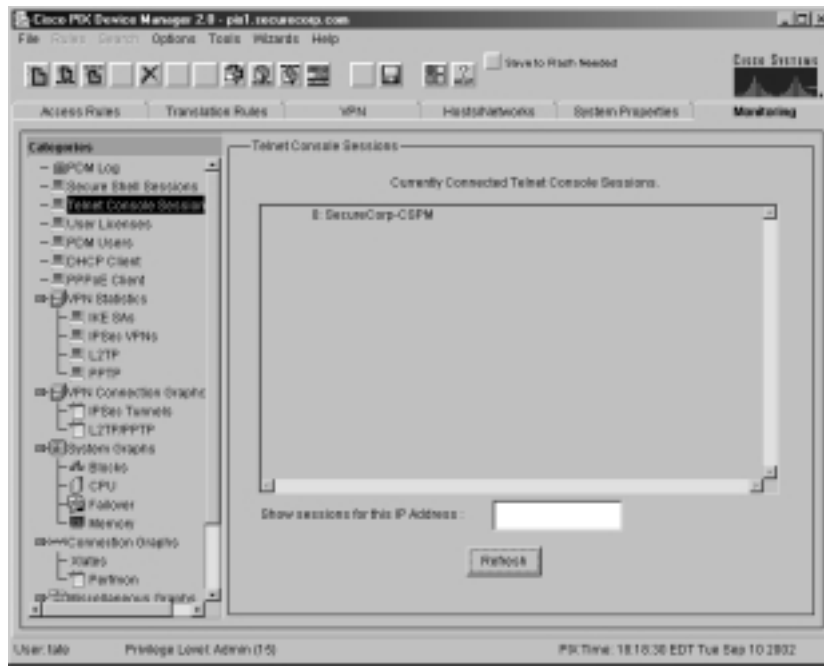
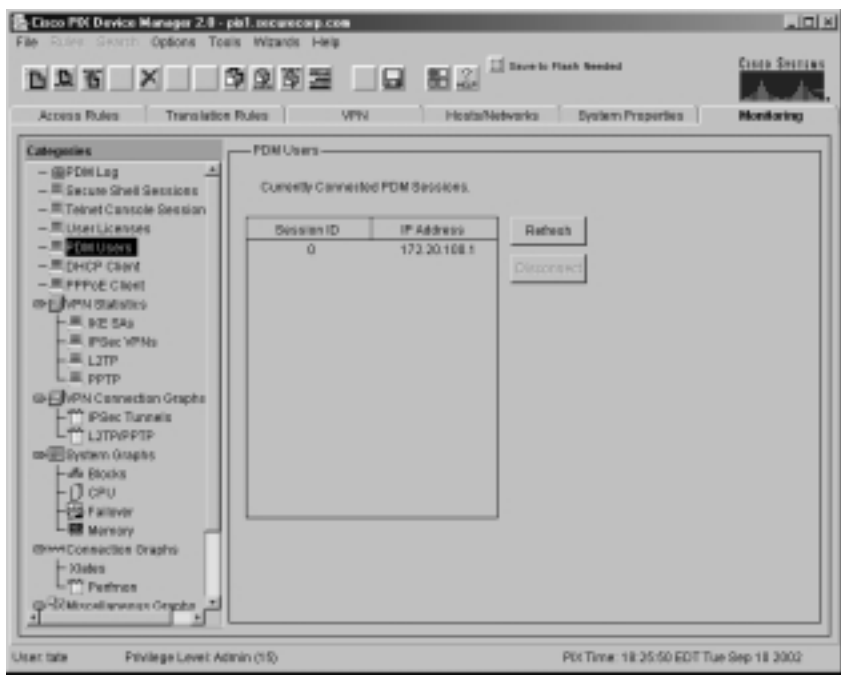


Figure 9.79 The PDM Users Screen



In this example, one PDM session is currently enabled from network host 172.20.100.1.

Sessions can also be disconnected in real time. For instance, if you suspect the PDM session shown is unauthorized, highlight the session and click the **Disconnect** button. Similarly, disconnect functionality is available from the Secure Shell Sessions category as well.

PDM provides the ability to view the current user license count on the firewall. This functionality is especially important for small organizations and SOHO environments that have limited license PIX firewalls. Click the **User Licenses** category to view the currently used licenses. The screen displays two values: Number of Licenses in Use and Number of Licenses Available. Click the **Refresh** button to redraw the screen with the most current user license statistics.

The categories DHCP Client and PPPoE Client both show statistical information regarding the PIX firewall's client DHCP and PPPoE services. These two categories only have relevant information if the external interface of the PIX firewall is configured with either DHCP or PPPoE client services.

If so, the assigned IP address, subnet mask, server IP address, lease time information, default gateway IP address, and other related information can be found by clicking these categories. These categories are especially helpful for small organizations and SOHO environments whose firewalls have dynamic settings.

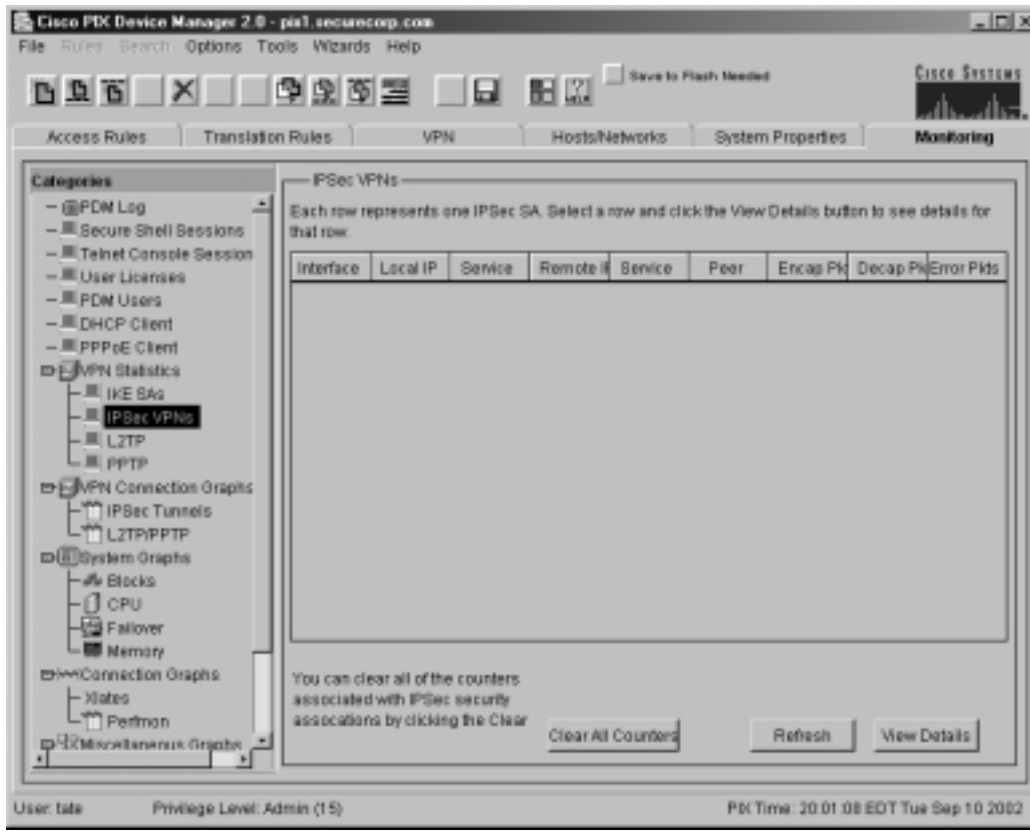
From the VPN Statistics category, administrators can gain valuable information regarding active IKE security associations (SAs) and the various VPN protocols available on the PIX firewall. Four specific subcategories exist under the VPN Statistics category:

- IKE SAs
- IPSec VPNs
- PPTP
- L2TP

Information for each of these subcategories is only available if active VPN sessions exist on the firewall. Each of the VPN Statistics subcategories includes the ability to manually refresh the screen view for updated VPN-related statistics. Furthermore, each subcategory except IKE SAs facilitates detailed information via the View Details button.

For example, the IPSec VPNs screen provides the ability to view VPN source and destination IP information, including packet encapsulation, decapsulation, and error counts. The IPSec VPNs screen is shown in Figure 9.80.

Figure 9.80 The IPSec VPNs Screen



The IPSec VPNs, L2TP, and PPTP subcategory screens allow you to monitor various metrics regarding active VPN connections and refresh the statistics data manual when required. You can also reset the metric counts by clicking the **Clear All Counters** button. Highlight a connection and click the **View Details** button to view detail regarding a specific VPN.

Graphs

The remaining categories on the Monitoring tab pertain to performance-related graphs on the PIX firewall. Monitoring graphs are grouped into five categories:

- VPN Connections Graphs
- System Graphs
- Connection Graphs

- Miscellaneous Graphs
- Interface Graphs

Within each of these categories are several subcategories with various options and details. These categories and their respective subcategories are discussed in this section.

All graphs, regardless of functionality or purpose, are configured using the same methodology. For instance, when you click a specific subcategory under VPN Connections Graphs, a list of available graphs for that subcategory appears in the Available Graphs for: field. To select a specific graph, highlight it and click the **Add** button to transfer to the graph name to the Selected Graphs(s) field. To create a new graph with the elements added to the Selected Graphs(s) field, type a new descriptive name in the **Graph Window** drop-down list. Finally, click the **Graph it** button to create a new browser window with the new graphs.

NOTE

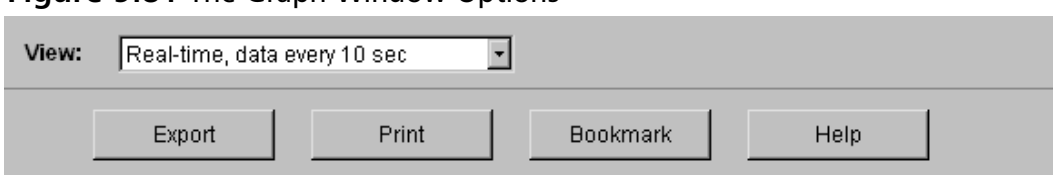
You can configure up to four graphs in a single window; these graphs can be from different categories or subcategories.

Alternatively, you can select an existing graph from the **Graph Window:** drop-down list and modify its configuration.

Once the graphs appear in a new browser window, identical options are available for each. The data in each graph window can be displayed in two ways: graphically or in tabular format. To alter the data display, click either the **Graph** or **Table** tab at the top of the graph window.

Furthermore, several options exist in the graph window, including four buttons and a drop-down menu, as shown in Figure 9.81.

Figure 9.81 The Graph Window Options



Various time ranges are available for graphs displayed in PDM. These can be selected by clicking the **View** drop-down menu and selecting an option shown in Table 9.6.

Table 9.6 Graphical Update Options

Data Window	Update Frequency
Real time	Every 10 seconds
Last 10 minutes	Every 10 seconds
Last 60 minutes	Every 1 minute
Last 12 hours	Every 12 minutes
Last 5 days	Every 2 hours

Each of these options alters the appearance of the graph accordingly.

NOTE

You must enable PDM History Metrics from the System Properties tab History Metrics category before you use the Monitoring tab to view graphs.

In addition to altering the graph appearance, you can also export the graph data to a comma-delimited text file, print the graph, or save the graph URL as a bookmark in your browser for future use. To do so, click the appropriate **Export**, **Print**, or **Bookmark** button from the graph window.

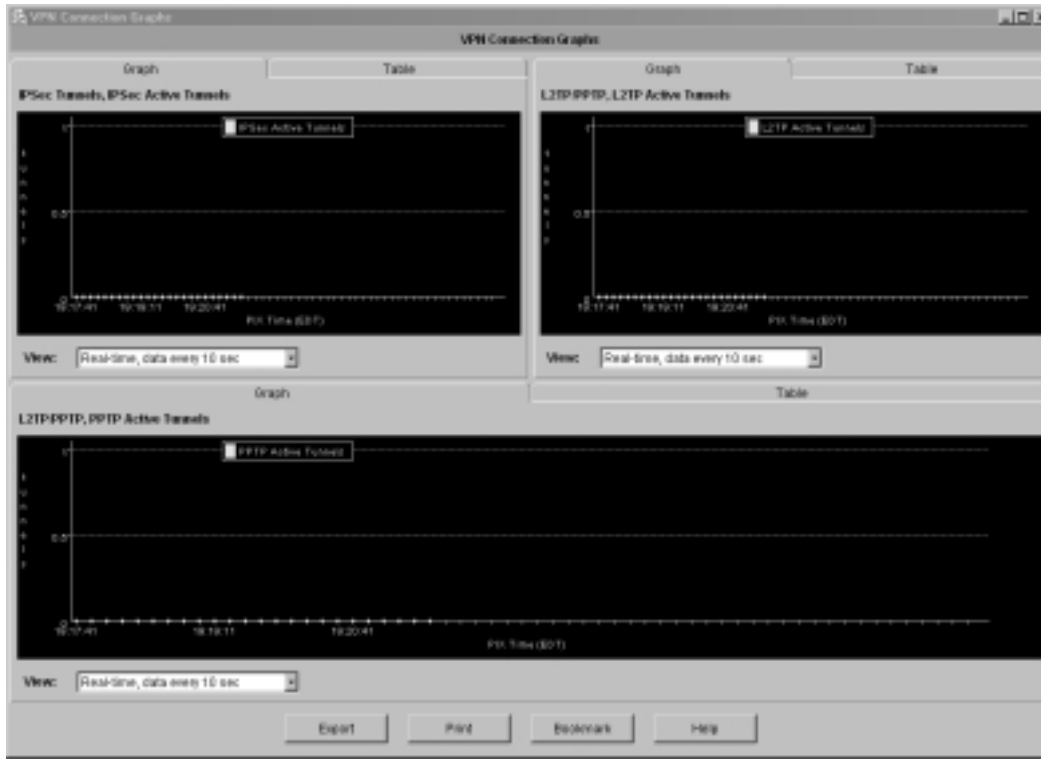
Now let's look at the various graphs and some examples of using the PDM graphs in the real world.

VPN Connection Graphs

The first group of graphs available through PDM is related to VPN connections. From the Monitoring tab, click the **VPN Connection Graphs** category to view the subcategories IPSec Tunnels and L2TP/PPTP. Each of these subcategories offers multiple graphs particular to the tunnel protocol represented.

The IPSec Tunnels subcategory includes graphs for IPsec active tunnels and IKE active tunnels. The L2TP/PPTP subcategory has graphs for L2TP and PPTP active tunnels and L2TP and PPTP active sessions. An example graph is included in Figure 9.82.

Figure 9.82 VPN Connection Graphs



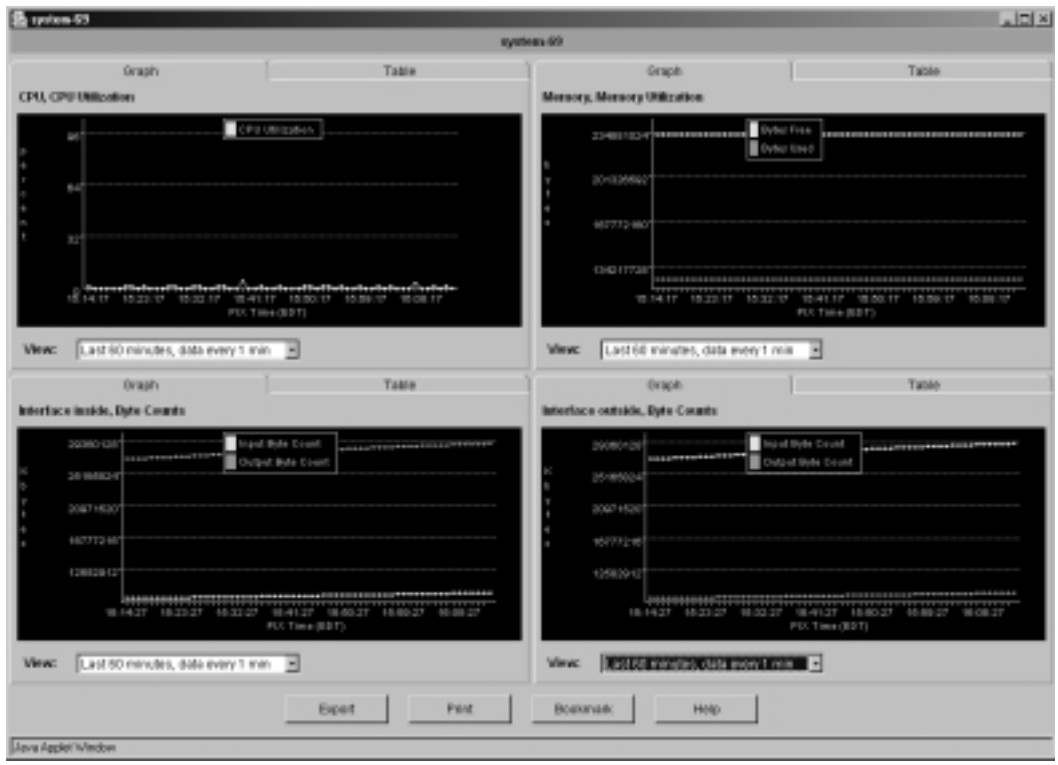
This graph depicts active IPSec, L2TP, and PPTP tunnels and would be extremely beneficial for administrators who terminate VPN connections on their PIX firewall. Using this graph, firewall administrators can see real-time information regarding VPN connections.

System Graphs

The next graph category available from the Monitoring tab is System Graphs. This category includes four subcategories: Blocks, CPU, Failover, and Memory. Each of these subcategories includes specific graphs that pertain to the system attributes they represent. For instance, the Blocks subcategory has two available graphs, Blocks Used and Blocks Free; the CPU and Memory Utilization subcategories each have only one graph regarding their utilization. The Failover subcategory includes several graphs, such as Translation Information, TCP Connection Information, and Xmit Queue.

The graphs available from the System Graphs category are generally useful to monitor and view performance variables regarding memory and CPU utilization. For example, the graph in Figure 9.83 depicts CPU Utilization, Memory Utilization, and Interface Byte Counts for both internal and external interfaces. We discuss interface graphs later in the chapter. The graphs shown in Figure 9.83 are updated every minute and show the last 60 minutes of information.

Figure 9.83 System Graphs



This graph is highly useful for seeing the 60-minute trends of resource allocation and utilization on the PIX firewall. To view longer-term historical trends, simply click the **View** pull-down list and select a longer time interval, such as **Last Five Days**.

Connection Graphs

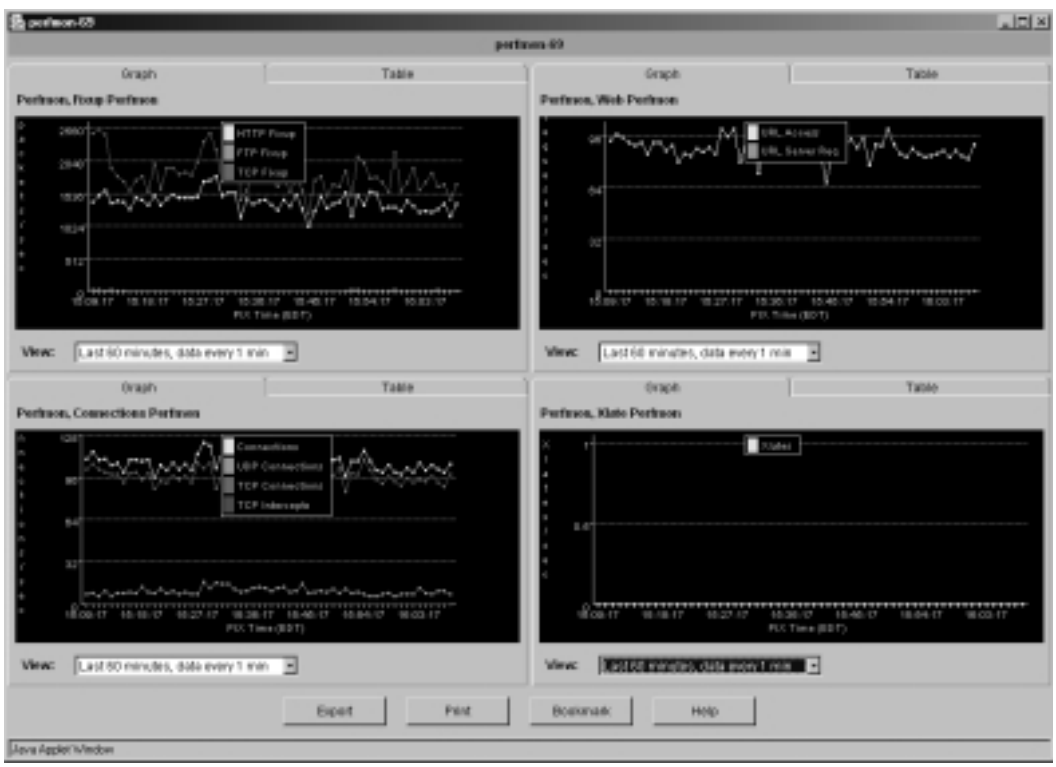
The Connection Graphs subcategory is next in the Monitoring tab. Connection graphs depict information regarding specific traffic through the firewall, such as Web or individual fixup traffic. From the Connection Graphs subcategory, two subcategories are available: Xlates and Perfmon.

The Xlates subcategory includes one available graph called Xlate Utilization. This graph shows the total address translations currently in use on the firewall. The Perfmon subcategory includes several graphs as follows:

- **AAA Perfmon** Displays the number of authentication, authorization, and accounting requests sent to an AAA server.
- **FixUp Perfmon** Displays the number of packets for traffic processed by the HTTP, FTP, or TCP fixup routines.
- **Web Perfmon** Displays the number of URL requests processed by the PIX firewall and the number of Websense requests made by the PIX firewall.
- **Connections Perfmon** Displays the total number of connections, TCP connections, UDP connections, and TCP intercepts processed by the PIX firewall.

Examples of connection graphs are included in Figure 9.84.

Figure 9.84 Connection Graphs

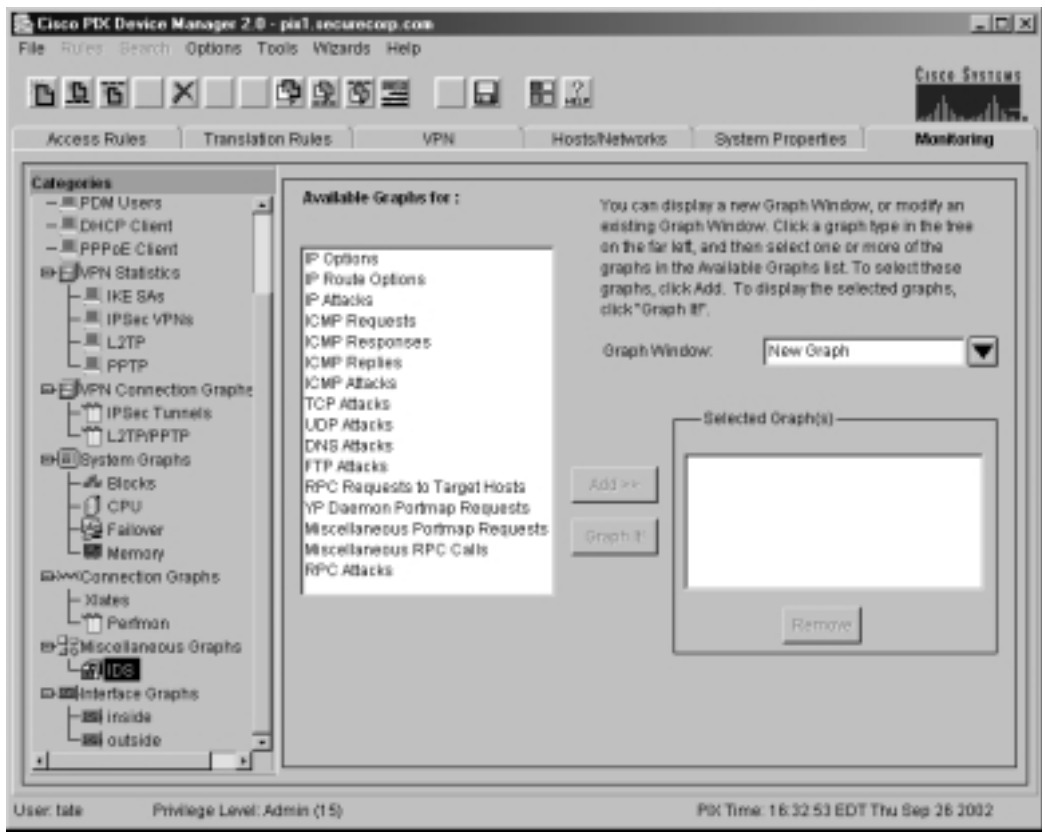


The graphs depict Fixup Perfmon, Web Perfmon, Connection Perfmon, and Xlate Perfmon. These graphs can help firewall administrators understand traffic trends and particular I/O through the firewall. In the examples shown in Figure 9.82, the firewall is running in a no-NAT configuration. This is evident based on the amount of traffic in the Xlate Perfmon graph (there is none!).

Miscellaneous Graphs

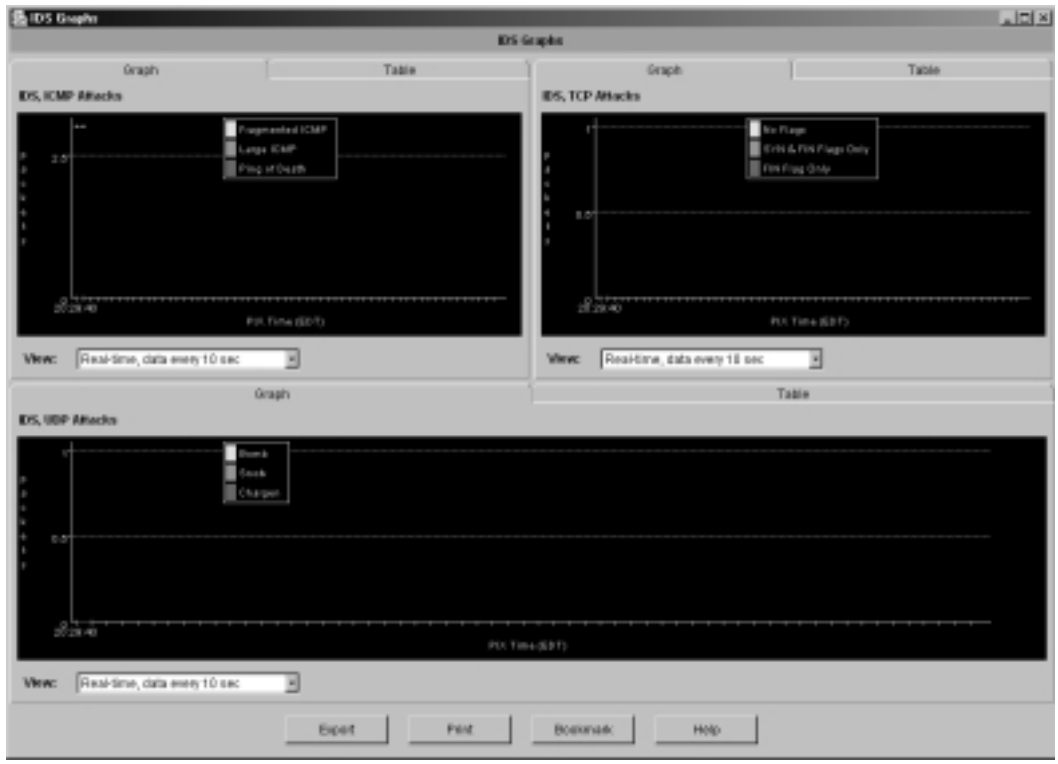
The Miscellaneous Graphs category includes the IDS subcategory. This subcategory can provide information related to the various IDS capabilities imbedded on the PIX firewall. As shown in Figure 9.85, 16 different graphs are available from the IDS subcategory.

Figure 9.85 Miscellaneous Graphs Setup



Using the IDS graphs, you can monitor in real time potential threats to your network. For instance, Figure 9.86 depicts ICMP, TCP, and UDP attacks graphically and updates every 10 seconds.

Figure 9.86 IDS Graphs



In this instance, no attacks have been detected on the PIX firewall. As you can see, various attack vectors are depicted in each graph specific to the protocol represented.

Interface Graphs

The final category of graphs available from the Monitoring tab in PDM is Interface Graphs. A subcategory representing each active interface on the PIX firewall appears in the Interface Graphs category. From each specific interface subcategory, 10 graphs are available:

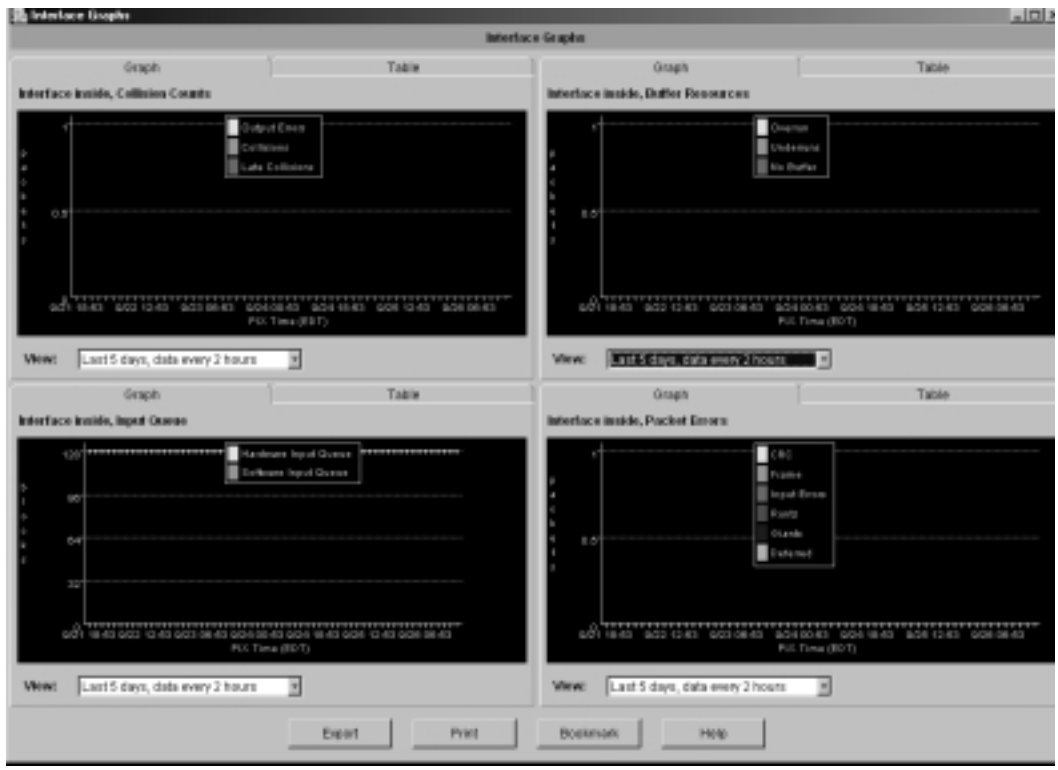
- Packet Rates
- Bit Rates
- Byte Counts
- Packet Counts

- Buffer Resources
- Packet Errors
- Miscellaneous (Received Broadcasts)
- Collision Counts
- Input Queue
- Output Queue

Each of these graphs can be immensely helpful in troubleshooting performance issues or misconfigurations such as duplex mismatches, physical cabling issues, or port negotiation problems.

For instance, if you believe you are experiencing latency with traffic passing through the PIX firewall, you could construct a set of graphs such as the one shown in Figure 9.87.

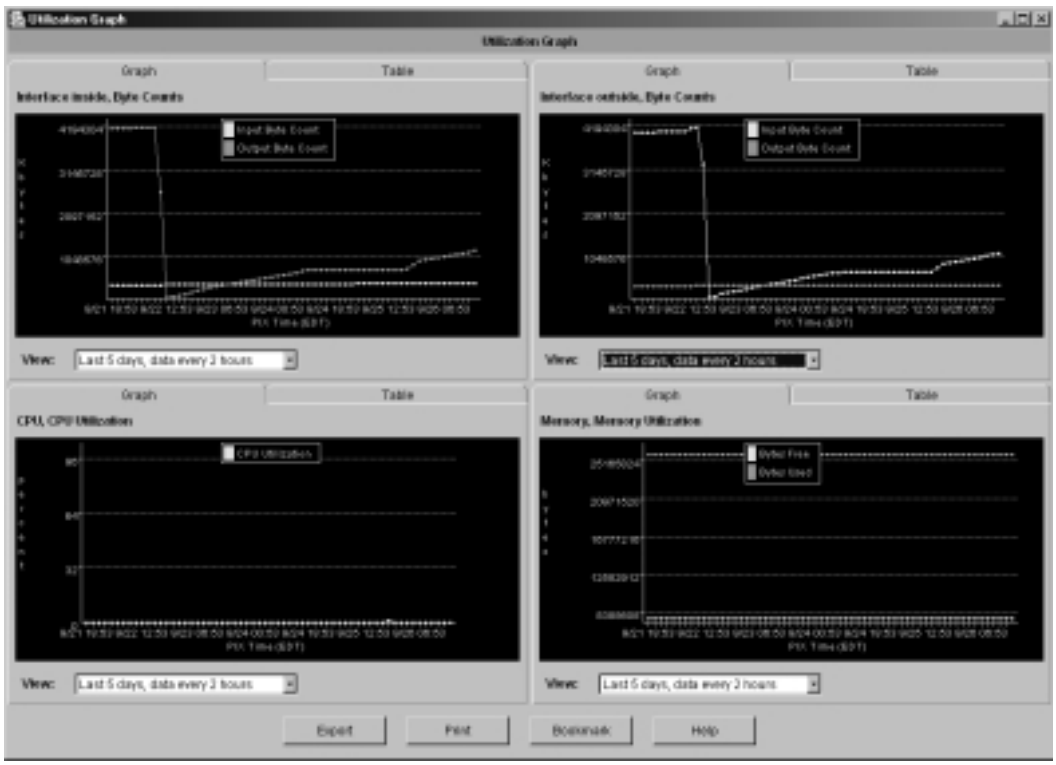
Figure 9.87 Interface Graphs



With this set of graphs, you can visually depict the total number of collisions, buffer resource overruns, input queue blocks used, and various packet errors for the past five days. If you are seeing high buffer overruns and collision counts, the interfaces on the PIX might be saturated with traffic. Perhaps the PIX memory is at 100 percent utilization. Alternatively, high packet errors for various attributes such as runts or input errors could signal physical cabling issues.

To verify possible causes, you could use this set of graphs with additional graphs such as those depicted in Figure 9.88.

Figure 9.88 Utilization Graphs



This combination of graphs shows interface byte counts for both interfaces as well as CPU and memory utilization over a period of five days. These graphs, in combination with the previously shown interface graphs, can help diagnose various problems associated with PIX performance.

You have mastered the power of PIX monitoring, so now let's look at some connectivity control mechanisms available through the Monitoring tab and the PIX CLI.

Monitoring and Disconnecting Sessions

Several CLI commands are available to monitor and disconnect administrative sessions. To monitor and disconnect PDM sessions, use the *show pdm* sessions command. This command displays all active PDM sessions, including the session IDs and the PDM clients' IP addresses. To disconnect a PDM session, use the *pdm disconnect <session_id>* command, where the *session_id* refers to the identification number listed in the *show pdm sessions* command.

You can also use the *clear pdm* command to remove all PDM locations, disable PDM logging, and clear the PDM internal buffer. Although the *clear pdm, pdm history, pdm location,* and *pdm logging* commands appear in your configuration and are available through the CLI, they are intended as internal PDM-to-PIX firewall commands accessible through PDM.

Summary

As you have seen in this chapter, PDM is a highly capable graphical interface for managing the PIX firewall. In addition to providing nearly all CLI functionality, PDM includes several features to further simplify the ongoing maintenance and operations firewall administrators and security policy makers perform. Because PDM is Java based and runs as a signed applet over an SSL-encrypted browser session, administrators can use it securely from any authorized client. This remote management capability can be highly valuable in large, distributed environments.

Of the vast PDM functionality, perhaps most powerful are the PDM wizards, which include the Startup Wizard and the VPN Wizard. Using these tools, administrators are guided using interactive prompts through the often-complex process of building PIX configurations and VPN tunnel services.

In addition to the wizard functionality, PDM facilitates full configuration of PIX firewall access, AAA, filter, NAT rules, logging, user accounts, and IDS configurations. This functionality includes the ability to manage complex, grouped services and network objects, which is new functionality in the PIX firewall software.

The PDM GUI is intuitive and well organized and helps prevent accidental syntax and configuration errors that could cause the firewall to fail. Moreover, PDM can be used as a CLI learning tool for administrators who are not completely proficient with the PIX firewall command line by previewing all commands sent to the PIX.

PDM also includes powerful real-time graph and reporting functionality. This tool helps firewall administrators understand the historical and current performance and functionality of the PIX. Furthermore, the IDS graphical reporting available through PDM can provide important insight into the potential security risks posed to organizations.

Whether you are managing a single PIX firewall, five redundant PIX pairs, or 50 corporate firewalls, PDM is a handy and powerful tool for firewall administrators.

Solutions Fast Track

Features, Limitations, and Requirements

- ☑ PDM 2.1 is supported on all PIX 501, PIX 506/506E, PIX 515/515E, PIX 520, PIX 525, and PIX 535 platforms running PIX firewall software version 6.2 or higher as well as the FWSM 1.1.
- ☑ Some CLI commands reduce PDM functionality to monitor-only mode.
- ☑ PDM is a signed Java applet downloaded to the client machine through a compliant browser. Therefore, PDM is available from any compliant and authorized client workstation for firewall management.

Installing, Configuring, and Launching PDM

- ☑ You must acquire and install a Data Encryption Standard (DES) or 3DES activation key on the PIX before PDM will function.
- ☑ PDM can be installed on the PIX firewall in a process similar to that of a PIX software image upgrade.
- ☑ You can authorize specific IP addresses or networks for access via PDM using the *http* command.

Configuring the PIX Firewall Using PDM

- ☑ Administrators can use the VPN Wizard to build IPsec, L2TP, and PPTP tunnels.
- ☑ Object groups for services or network entities can be created and managed using PDM on the PIX firewall.
- ☑ Use the Reset PIX to the Factory Default Configuration option from the File drop-down menu on the PIX 501 and 506 platforms to return the PIX firewall to its original configuration.
- ☑ Rule sets can easily be rearranged from the Access Rules tab using the cut-and-paste functionality of the PDM Rules drop-down menu, the toolbar buttons, or the right-click mouse menu.

- ☑ To set up a syslog logging host, use the Logging category available from the PDM System Properties tab.

Monitoring the PIX Firewall Using PDM

- ☑ Administrators can permit monitor-only access to corporate officers or other VIP users so that they may view historical and current performance data on the PIX firewall.
- ☑ Real-time IDS events and performance data can be displayed using the monitoring functionality of PDM.
- ☑ Administrators can perform advanced troubleshooting techniques using the various monitoring graphs such as interface and system graphs.
- ☑ Administrative access (Telnet, SSH, and PDM sessions) can be monitored using PDM.
- ☑ SSH and PDM sessions can be terminated in real time through the PDM monitoring functionality.
- ☑ VPN connections, including IPsec, L2TP, and PPTP tunnels, are available for monitoring via the VPN Connection Graphs category from the PDM Monitoring tab.
- ☑ To view monitoring statistics with PDM, you must first enable History Metrics from the System Properties tab.
- ☑ Up to four graphs from multiple categories can be grouped together for a more comprehensive visual representation of PIX firewall metrics.

Monitoring and Disconnecting Sessions

- ☑ Use the *show pdm sessions* and *show ssh sessions* commands to view real-time administrative connections to the firewall.
- ☑ To view active PDM sessions, use the *show pdm sessions* command.
- ☑ To terminate active PDM sessions, use the *pdm disconnect <session_id>* command.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: Can I monitor and manage remote PIX firewalls using PDM from a central facility or other offsite locations?

A: Yes. Using the *http* command via the CLI or PDM, you can authorize an IP range or a specific IP address for access to PDM. The PDM connection is encrypted for security.

Q: Can I set up AAA for administrative connectivity to the PIX firewall using PDM?

A: Yes. PDM includes full AAA configuration functionality. Additionally, you can use PDM to configure the PIX for AAA services for PDM itself.

Q: Can I use PDM to disconnect a user connected to the PIX firewall via Telnet?

A: No. Currently, the disconnect feature is only available for PDM and SSH sessions.

Q: Do I need a special license to enable PDM on my PIX firewall?

A: Yes. You need a DES or 3DES activation key from Cisco before PDM will function properly. A 56-bit DES key is available free. The 168-bit 3DES key is available from Cisco at an additional cost.

Q: Does PDM include VPN maintenance functionality?

A: Yes. VPN maintenance functionality is available in PDM. Additionally, PDM includes VPN functionality not present in the CLI, such as the VPN Wizard.

Q: Can I use PDM to manage multiple PIX firewalls at once?

A: Yes, but a separate instance of PDM must be launched for each firewall.

Chapter 10

Troubleshooting and Performance Monitoring

Solutions in this chapter:

- **Troubleshooting Hardware and Cabling**
- **Troubleshooting Connectivity**
- **Troubleshooting IPsec**
- **Capturing Traffic**
- **Monitoring and Troubleshooting Performance**

☑ **Summary**

☑ **Solutions Fast Track**

☑ **Frequently Asked Questions**

Introduction

This chapter focuses on troubleshooting PIX firewalls. Once you have mastered its command syntax and basic firewall operations, the PIX is a relatively simple device to configure. Its library of commands is small compared to that of Cisco routers and switches. In previous chapters, we covered the PIX firewall in detail, from the various models in the product line to simple and advanced configurations. This book contains information on how to integrate the PIX firewall into your existing network. As good as your PIX configuration is, problems will still crop up, and you need to know how to resolve them. The purpose of this chapter is to present a methodology that you can use to attack these problems and avoid missing critical troubleshooting steps.

Hardware and cabling problems can be a bane to an otherwise well-functioning network. A hardware problem becomes apparent if you know which indicators to monitor. The limited number of cable types that the PIX supports eases our cable troubleshooting considerably. This chapter provides technical information about these cables so you can validate them.

The PIX firewall is an IP device. Granted, it is a highly specialized device that performs vital security functions, but it is still an IP device. As such, it needs to know where to send traffic. We highlight some common connectivity problems and how you can address them. A valuable function of the PIX firewall is its ability to conserve IP address space and hide network details via Network Address Translation (NAT). If you have problems with NAT, you must be able to isolate and eliminate them.

The PIX firewall provides several access control mechanisms, from simple access lists to complex conduit statements. These access mechanisms have simultaneous loose/tight properties in that certain traffic is allowed while other traffic is denied. Your troubleshooting will not only seek to resolve access problems but also find the right balance between permitting and denying traffic.

Entire books have been written on IPsec, and for good reason. IPsec can protect your traffic from end to end without having to be implemented at every hop along the way. IPsec configuration can be complex. You must be intimately familiar with IPsec operations in order to support and troubleshoot it. This chapter covers several key aspects of IKE and IPsec to aid your monitoring and support.

Capturing network packets on the PIX firewall can enable you to troubleshoot more effectively. The PIX firewall offers several features that you can use to capture traffic for analysis and problem isolation. Available tools include native PIX commands as well as third-party tools for network capture and packet decode.

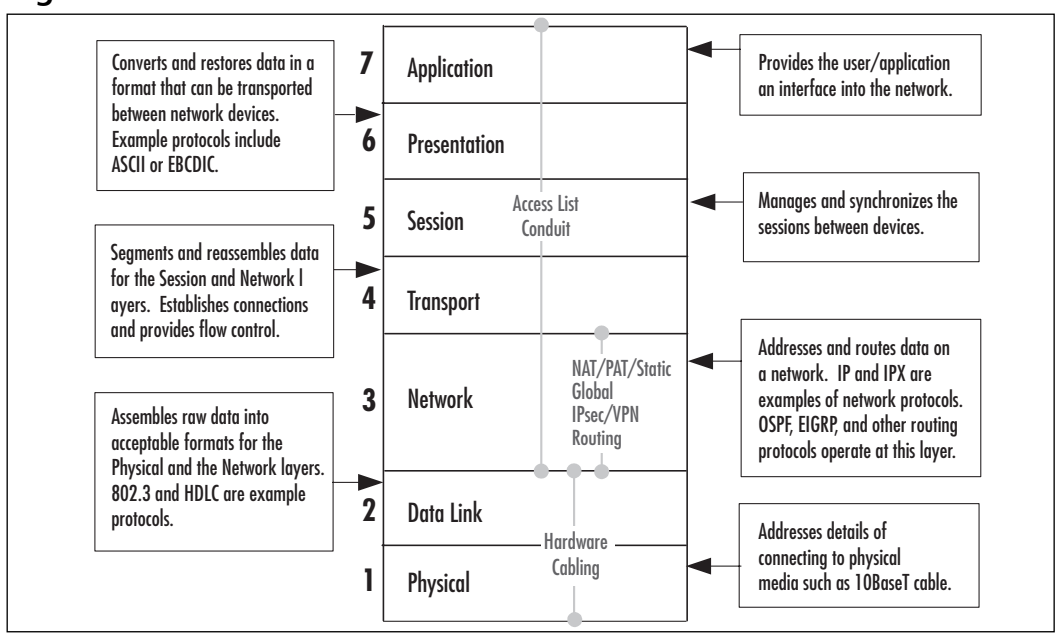
How do you know if your PIX firewall is performing as well as it should? How would you know if it was overloaded? You need to monitor firewall performance and health proactively. The goal of monitoring is to prevent minor glitches from turning into major problems. The output of your monitoring efforts can be quite dense and arcane, so you need to know how to interpret what you are monitoring.

Troubleshooting Hardware and Cabling

The most important thing to remember in troubleshooting is to tackle your problems logically so you don't miss any important components or steps. You must confirm the health of all the components that make up the firewall. When addressing PIX firewall problems, you would be best served using the OSI model to guide your efforts. This model was created to guide development efforts in networking by dividing functions and services into individual layers. Per the OSI model, peer layers communicate with each other. For example, the network layer at one host communicates with the network layer at another host.

The approach advocated in this chapter is based on the OSI model shown in Figure 10.1. Problems are tackled starting at the lowest layer, such as validating hardware and cabling at the physical layer. Only when the components at the lower layer have been validated do you turn your attention to components at a higher layer.

Figure 10.1 The OSI Model



This chapter organizes troubleshooting efforts by the OSI model. Initial troubleshooting starts at Layer 1, the physical layer. Once all physical components have been validated, the troubleshooting focus is shifted to the data link layer components, and so on, up the OSI stack. This controlled approach ensures that we do not miss any facet of our security configuration where the problem could be.

Our first steps in troubleshooting start with physical layer issues. In the context of the PIX firewall, physical components include the firewall hardware and cabling. We start our discussion with a quick overview of the PIX firewall hardware architecture and cabling.

Troubleshooting PIX Hardware

Knowing the details of each PIX firewall model can be helpful in validating your configuration and troubleshooting. Such knowledge can quicken your problem-solving process from the onset by enabling you to determine how to interpret the symptoms you are witnessing. If you use the wrong firewall model for the wrong function, no amount of troubleshooting is going to make it work.

It can be said that your troubleshooting actually starts with your network design and security planning. There are several models of the PIX firewall, each capable of supporting certain numbers and types of network interfaces. Each model has its own upper limit on the number of maximum simultaneous connections, as shown in Figure 10.1. The specific models were discussed at length in Chapter 2, so in Table 10.1 we provide only a snapshot of each model.

Table 10.1 PIX Firewall Model Features and Capabilities

Model	Interface Types Supported	Maximum Number of Interfaces	Failover Support
501	Ethernet Fast Ethernet	Fixed 10BaseT Four-port 10/100 switch	No
506 End of Sale	Ethernet Fast Ethernet	Two fixed 10/100 Ethernet	No
506E	Ethernet Fast Ethernet	Two fixed 10/100 Ethernet	No
515 End of Sale	Ethernet Fast Ethernet	Two fixed 10/100 Ethernet Two expansion slots Maximum: Six ports	Yes

Continued

Table 10.1 Continued

Model	Interface Types Supported	Maximum Number of Interfaces	Failover Support
515E	Ethernet Fast Ethernet	Two fixed 10/100 Ethernet Two expansion slots Maximum: Six ports	Yes
520 End of Sale	Ethernet Fast Ethernet	Two fixed 10/100 Ethernet Six interface slots Maximum: Six ports	Yes
525	Ethernet Fast Ethernet Gigabit Ethernet	Two fixed 10/100 Ethernet Four interface slots Maximum: Eight ports	Yes
535	Ethernet Fast Ethernet Gigabit Ethernet	Nine interface slots Maximum: 10 ports	Yes

The "E" at the end of certain models indicates a faster processor and wider backplane, meaning the firewall can handle greater traffic loads. Failover is supported only on PIX firewall models 515 and up, something you need to remember in your planning.

It is important to know whether the PIX firewall you are using is adequate for the demands planned for it. For example, if you have a network on which 100,000 simultaneous connections will be requested through the firewall and you are using a PIX 501, the firewall will immediately become congested and be virtually unusable. In this scenario, no amount of troubleshooting and configuration will enable the PIX 501 to support the load. The capacity of each firewall model is important because it determines the load that can be placed on that firewall. Overloading your firewall is an invitation to crashes or congestion. Underloading a PIX firewall, although great for performance, can be wasteful in terms of unused capacity and monetary return on investment. For example, if you have a network on which there will never be more than 200 simultaneous connections, installing a PIX 535 means that you will not recoup your hardware or software investment, although performance will be fantastic.

The different models support different types of interfaces and in specific quantities, as shown in Table 10.1. Not shown in the table is the fact that Token Ring and FDDI are also supported by several of the models. Cisco ceased PIX firewall support for Token Ring and FDDI networks, starting with PIX software version 5.3. As a rule of thumb, do not mix and match interfaces: Configure the PIX firewall as all Token Ring, all Ethernet, or all FDDI. Maintaining such

network purity reduces the burden on the PIX firewall since it will not have to translate between the different LAN formats. Only models 515 up and support interfaces other than Ethernet.

The PIX firewall has a system for identifying its network interfaces, which you need to understand in order to troubleshoot the right piece of hardware. Not knowing how interfaces are enumerated and identified can consume valuable time that could otherwise be used for troubleshooting. Figure 10.2 shows how to "read" the network interface identification scheme. Interface card numbering starts with 0 at the right, with card slot numbers increasing as you go left. The slot in which the card is installed determines the number that is given to that card. Modular ports are numbered sequentially starting at the top, then left to right, starting with 0 for the port at the left of the topmost card.

For example, the leftmost port on an Ethernet interface card installed in Slot 2 would be identified as Ethernet 10. Fixed interfaces are first numerically starting on the right at 0, then the next fixed interface to the left is 1. The first installed network interface card port would be Ethernet 2. It is important that you learn this scheme not only to identify the specific cards but to also ensure that your configuration and troubleshooting efforts focus on the correct interface.

The memory architecture of the PIX firewall is somewhat similar to that of Cisco routers with the exception that there is no NVRAM memory. The PIX uses flash memory to store the firewall operating system (image) as well as the configuration file. Main memory is used to handle data being processed. As a rule of thumb, the flash memory should be big enough to hold the software image and the configuration. Of all the memory types, main memory can potentially have the most significant impact on performance since it is the working space of the firewall. Main memory is used to store data that is waiting to be processed or forwarded. You can never have too much, and you will definitely notice when you have too little, because packet loss will increase or IPsec traffic will become lossy or laggardly.

Each firewall has visual indicators of operation in the form of light-emitting diodes (LEDs). These LEDs vary by model, but some are common to all. Figure 10.3 shows several PIX firewall LEDs and their meanings. Nurturing your knowledge of these LEDs will enable you to start your Layer 1 troubleshooting from the outside.

Figure 10.2 PIX Firewall Interface Numbering

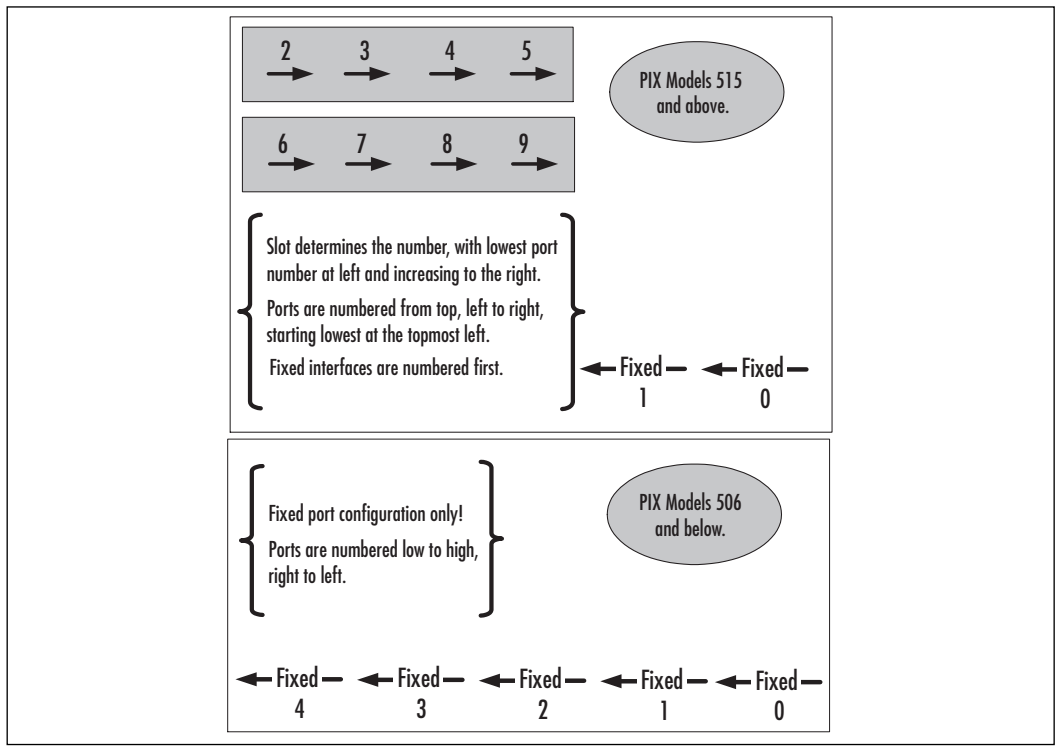
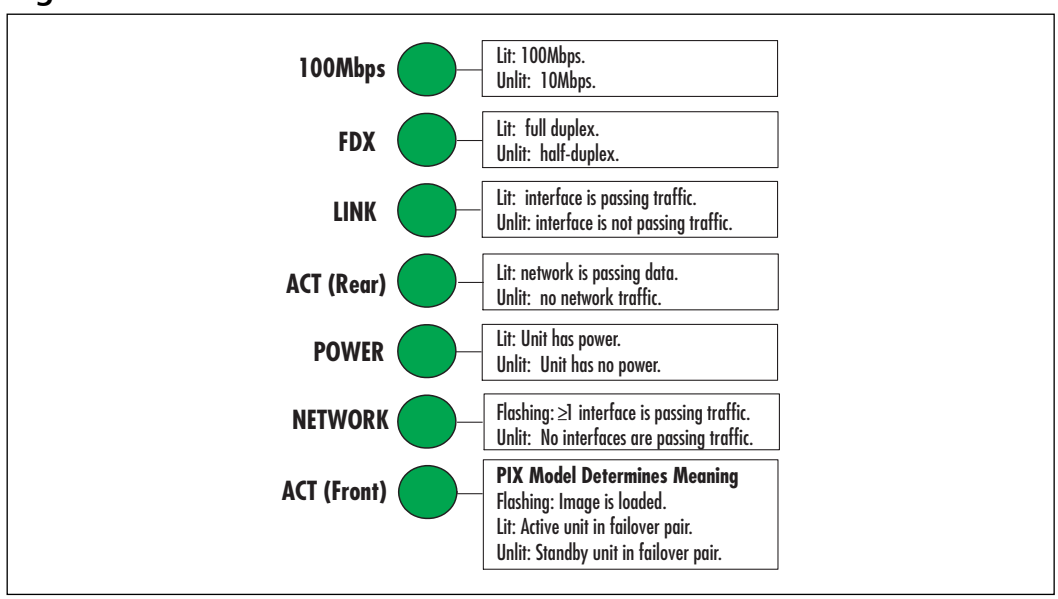


Figure 10.3 PIX Firewall LED Indicators



Study the information in Figure 10.3. The LEDs can be lit, unlit, or flashing, all of which indicate specific conditions. The ACT LED, since it can appear on both the front and rear of the PIX, deserves special attention. On certain models, such as the PIX 506 and 506E, the front LED flashes to indicate that the PIX software image has been loaded. When you're troubleshooting, this indicator would be sufficient to tell you if your software image has been loaded correctly or not at all. On higher-end models such as the 515 and up, the same LED indicates which PIX firewall is active and which is standby in a failover pair. This information can be very useful in determining if your failover configuration is cabled correctly.

During the PIX boot sequence, the power-on self-test (POST) can provide a wealth of information to help determine from the onset whether the PIX firewall is healthy or ill. We use an example boot sequence (see Figure 10.4) to guide our discussion.

Figure 10.4 PIX Firewall Bootup

```
CISCO SYSTEMS PIX-501
Embedded BIOS Version 4.3.200 07/31/01 15:58:22.08
Compiled by morlee
16 MB RAM

PCI Device Table.
Bus Dev Func VendID DevID Class                Irq
 00  00  00   1022   3000  Host Bridge
 00  11  00   8086   1209  Ethernet             9
 00  12  00   8086   1209  Ethernet             10

Cisco Secure PIX Firewall BIOS (4.2) #6: Mon Aug 27 15:09:54 PDT 2001
Platform PIX-501
Flash=E28F640J3 @ 0x3000000

Use BREAK or ESC to interrupt flash boot.
Use SPACE to begin flash boot immediately.
Reading 1536512 bytes of image from flash.
############################################################################
16MB RAM
Flash=E28F640J3 @ 0x3000000
```

Continued

Figure 10.4 Continued

```
BIOS Flash=E28F640J3 @ 0xD8000
mcwa i82559 Ethernet at irq  9  MAC: 0008.e317.ba6b
mcwa i82559 Ethernet at irq 10  MAC: 0008.e317.ba6c
------------------------------------------------------------
                                  ||         ||
                                  ||         ||
                                 ||||       ||||
                              ..:||||||:..:||||||:..
                             c i s c o S y s t e m s
                            Private Internet eXchange
  ------------------------------------------------------------
                            Cisco PIX Firewall

Cisco PIX Firewall Version 6.2(2)
Licensed Features:
Failover:             Disabled
VPN-DES:              Enabled
VPN-3DES:             Disabled
Maximum Interfaces: 2
Cut-through Proxy:  Enabled
Guards:               Enabled
URL-filtering:       Enabled
Inside Hosts:         10
Throughput:           Limited
IKE peers:            5
****************************** Warning ******************************
  Compliance with U.S. Export Laws and Regulations - Encryption.

<<  output omitted  >>
  ****************************** Warning ******************************

Copyright (c) 1996-2002 by Cisco Systems, Inc.

                  Restricted Rights Legend
```

Continued

Figure 10.4 Continued

```
<<  output omitted  >>
Cryptochecksum(unchanged): 38a9d953 0ee64510 cb324148 b87bdd42

Warning: Start and End addresses overlap with broadcast address.
outside interface address added to PAT pool
Address range subnet is not the same as inside interface
```

The boot sequence identifies the version of the PIX operating system loaded on firmware used to initially boot. In this example, it is 4.3.200. This is important to know because this is the OS that will be used if there is no software image in flash memory. Notice that the first line identifies the model of firewall—information that can be useful if you are checking the firewall remotely.

After the POST is complete, the software image installed in flash is loaded and takes over from that point, as indicated by the "Reading 1536512 bytes of image from flash" line. The PIX firewall runs its checksum calculations on the image to validate it. The OS in the firmware is also validated. This is a layer of protection against running a corrupted operating system. In Figure 10.4, the image loaded from flash memory recognizes two Ethernet interfaces present on this unit and displays the MAC addresses associated with them.

The boot display provides information about the PIX firewall hardware. Figure 10.4 shows that this particular unit has 16MB of main memory, something that can be a performance factor, as previously discussed. Other types of hardware such as interfaces (quantity and type) and associated IRQ information are identified as well.

Some very useful information about the features supported by this firewall can save you countless hours of frustration. For starters, the exact version of the operating system is identified—version 6.2(2), in this case. More important, the features supported by this firewall are clearly enumerated. For example, VPN-DES is supported, whereas VPN-3DES is not. This makes sense since we are looking at a low-end PIX 501 with a limited license for 10 hosts and 5 IKE peers. This firewall supports cut-through proxy and URL filtering.

The last few lines of the boot screen can highlight errors that the operating system encountered when it parsed the configuration file. You should study these messages and determine if and how you must fix them. In our example, we have several problems with the way we have allocated our IP addresses. We also know that the outside interface address is now part of the PAT pool, which is something that we might or might not want, depending on our particular situation.

Once the firewall has completed booting, you can continue your hardware verification efforts using commands provided by Cisco. These are several commonly used commands to check the composition and health of your PIX firewall at Layer 1. Figure 10.5 illustrates the *show version* command, which provides a quick snapshot of your PIX firewall. Information provided by this command includes interface information, serial numbers, and so on, as shown in the command output in Figure 10.5. Use this command when you need information about your firewall's software and hardware. Some of the output is similar to what you saw during the boot sequence.

Figure 10.5 The show version Command

```
PIX1> show version

Cisco PIX Firewall Version 6.2(2)
Cisco PIX Device Manager Version 2.1(1)

Compiled on Fri 07-Jun-02 17:49 by morlee

PIX1 up 23 secs

Hardware:   PIX-501, 16 MB RAM, CPU Am5x86 133 MHz
Flash E28F640J3 @ 0x3000000, 8MB
BIOS Flash E28F640J3 @ 0xfffd8000, 128KB
0: ethernet0: address is 0008.e317.ba6b, irq 9
1: ethernet1: address is 0008.e317.ba6c, irq 10

Licensed Features:
Failover:            Disabled
VPN-DES:             Enabled
VPN-3DES:            Disabled
Maximum Interfaces: 2
Cut-through Proxy:  Enabled
Guards:              Enabled
URL-filtering:      Enabled
Inside Hosts:        10
Throughput:          Limited
IKE peers:           5
```

Continued

Figure 10.5 Continued

```
Serial Number: 406053729 (0x1833e361)
Running Activation Key: 0xc598dce8 0xf775fc1c 0xbd76cee8 0x3f41e74b
Configuration last modified by  at 06:28:16.000 UTC Thu Feb 7 2036
```

The first part of this command identifies the version of OS that is loaded and being used as well as the version of PIX Device Manager (PDM). Next in the output you see the amount of time that has elapsed since the unit was powered on. This information is useful because it can show if your PIX firewall was rebooted or power-cycled recently. The *show version* command gives additional details such as the model, amount of available memory, and CPU speed and type. It also tells you the amount of flash and BIOS memory. When troubleshooting, you should know this information in order to determine if the demands placed on the unit are reasonable. This unit has two Ethernet interfaces; notice that their MAC addresses are enumerated. The last part of the output provides the serial number of this unit as well as the activation key used to activate the image. Although it is not critical to troubleshooting, it might be necessary to provide this information to Cisco TAC should you need to call them for assistance.

When you're troubleshooting, the *show version* command should be one of the first (if not *the* first) commands that you execute to obtain a component inventory of the PIX firewall. It is especially vital that you know which features are supported by the firewall before you begin troubleshooting; otherwise, you could squander valuable time trying to determine why an unsupported featured is not working. When looking at the output of the *show version* command, ensure that you note the MAC addresses of the interfaces; this information can be useful in resolving Layer 2 to Layer 3 address-mapping issues.

The *show interface* command shown in Figure 10.6 is a tool that can provide information applicable to different layers of the troubleshooting process. It provides details on the network interfaces. As with Cisco routers, this command enables you to check the state of an interface and determine if it is operational. You can also see what each interface is labeled. This command and its associated output are discussed later in the chapter.

Figure 10.6 The show interface Command

```
interface ethernet1 "inside" is up, line protocol is up
  Hardware is i82559 ethernet, address is 0008.e317.ba6c
```

Continued

Figure 10.6 Continued

```
IP address 10.10.2.1, subnet mask 255.255.255.0
MTU 1500 bytes, BW 10000 Kbit full duplex
4 packets input, 282 bytes, 0 no buffer
Received 0 broadcasts, 0 runts, 0 giants
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
4 packets output, 282 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 babbles, 0 late collisions, 0 deferred
0 lost carrier, 0 no carrier
input queue (curr/max blocks): hardware (128/128) software (0/1)
output queue (curr/max blocks): hardware (0/1) software (0/1)
```

The output of the *show interface* command has useful applicability to the troubleshooting process. However, if you do not know how to read the output, the plethora of information presented will be of little value. One of the first things you need to determine with this command is if you want a particular interface to serve a particular network. In our example, Ethernet 1 is considered the "inside" network. As a part of our troubleshooting, we would ensure that Ethernet 1 is indeed connected to our "inside" network. The MAC address assigned to this interface is listed, as is the type of interface (Ethernet).

The maximum transmission unit (MTU) specifies the maximum packet size that this interface can pass without having to fragment it. Anything larger will be broken into the appropriate number of frames to enable passage through this interface. This can be an issue if you have devices that send large frames. This command also verifies the duplex operation of the interface; recall that the interface also has a full-duplex LED that you can use. Duplex mismatches between the PIX and LAN switches are a common problem and can be a headache. Ensure that the speed and duplex settings match on the PIX firewall and the switch.

There is a packet counter for inbound and outbound packets. This indicator tracks how many packets have transited this interface and the total number of bytes that these packets constituted. The "no buffer" counter is especially important to troubleshooting because it indicates the number of times that there were no buffers to store incoming packets until they could be processed by the CPU. If this counter increments, the interface is receiving more packets than it can handle. In this case, you need to upgrade to a higher-capacity interface or throttle back the incoming traffic. Each interface also has counters for tracking broadcasts and errors:

- **broadcasts** Packets sent to the Layer 2 broadcast address of this interface.
- **runts** Packets received that were less than Ethernet's 64-byte minimum packet size.
- **giants** Packets received that were greater than Ethernet's 1518-byte maximum packet size.
- **CRC** Packets that failed the CRC error check. Test your cables and also ensure there is no crosstalk or interference.
- **frame** Framing errors in which an incorrect Ethernet frame type was detected. Make sure you have the appropriate frame type configured on all your hosts.
- **overrun** Input rate exceeded the interface's ability to buffer.
- **ignored/abort** These counters are for future use. The PIX does not currently ignore or abort frames.
- **collisions** Number of transmitted packets that resulted in a collision. On a half-duplex interface, collisions do not necessarily indicate a problem, since they are a fact of Ethernet life.
- **underrun** Indicates that the PIX was too overwhelmed to get data fast enough to the network interface.
- **babbles** This is an unused counter. Babbles indicate that the transmitter has been on the interface longer than the time taken to transmit the largest frame.
- **late collisions** Collisions that occurred after the first 64 bytes of transmission. Unlike normal collisions, these indicate a problem. Usually late collisions are caused by faulty cabling, long cables exceeding specifiication, or an excessive number of repeaters.
- **deferred** Packets that had to be deferred because of activity on the link. This generally indicates a congested network since the interface has to keep backing off to find an available transmit window to send; this can become a perpetuating problem that consumes buffer space as outgoing packets have to be stored until a transmit windows opens.
- **lost carrier** The number of times the signal was lost. This can be caused by issues such as a switch being shut off or a loose cable.
- **no carrier** This is an unused counter.

NOTE

On a full-duplex interface, you should never see collisions, late collisions, or deferred packets.

The queue counters refer to the amount of data (measured in bytes) queued for reception and transmission. These counters provide a snapshot of what is currently queued at the time the command is issued. The queues will be depleted if the firewall receives more traffic than it can handle. When a packet is first received at an interface, it is placed in the input hardware queue. If the hardware queue is full, the packet is placed in the input software queue. The packet is then placed into a 1550-byte block (a 16384-byte block on 66MHz Gigabit Ethernet interfaces) and passed to the operating system. Once the firewall has determined the output interface, the packet is placed in the appropriate output hardware queue. If the hardware queue is full, the packet is placed in the output software queue.

In either the input or output software queue, if the maximum blocks are large, the interface is being overrun. If you notice this situation, the only way to resolve it is to reduce the amount of traffic or to upgrade to a faster interface.

Troubleshooting PIX Cabling

After you have ascertained that the PIX hardware is functional, your next step in troubleshooting should be to corroborate cabling. Unlike routers, which use a wide variety of cables, the PIX firewall has a relatively limited number of cable types that we care about in the context of troubleshooting: Ethernet and failover cables.

Certain models of the PIX firewall support Token Ring and FDDI networks in older software versions (up to version 5.3). Cisco has discontinued the sale of Token Ring and FDDI for PIX firewalls starting August 2001 and June 2001, respectively. Support is slated to cease in August 2006 and June 2006, respectively. We do not discuss Token Ring or FDDI cables in this book.

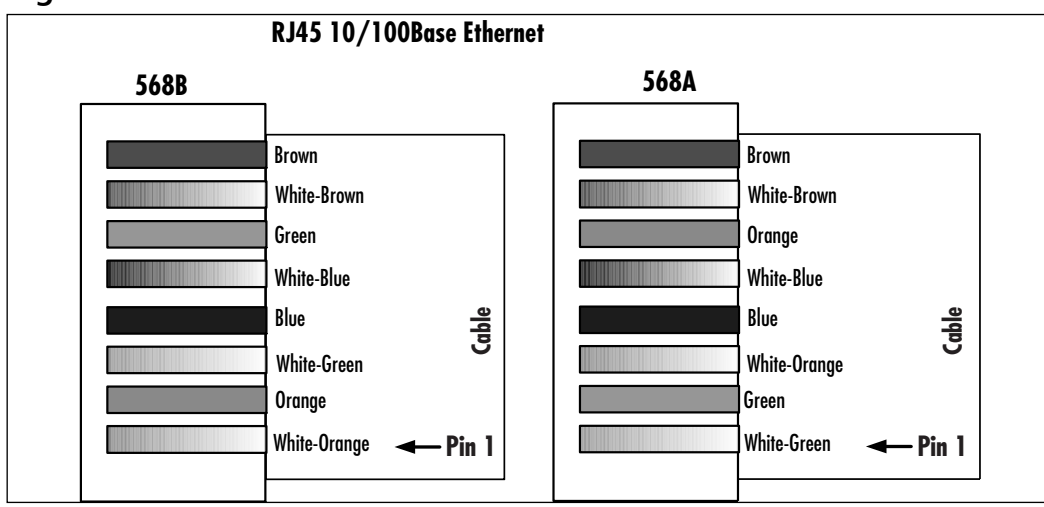
Regardless of the cables you are troubleshooting, you should adopt a structured approach. Table 10.2 summarizes some steps you should first take to check your cabling. Ensure that you perform these steps to avoid missing a minor cabling glitch that could be causing a major problem.

Table 10.2 Cable Troubleshooting Checklist

Problem	Troubleshooting Step
Correct cable connected to the correct interface?	Check cable and verify slot and port number.
Correct end of cable connected to correct interface?	*Failover cable only:* Primary end to the primary firewall and secondary end to the secondary firewall.
Correct cable type connected to equipment?	Cross cables, rollover cables, and so on to the correct ports.
Cable pinouts correct?	Visually inspect and check with cable tester.
Cable verified as good?	Test with a cable tester or swap with known good equipment and test.

All PIX firewalls support 10Mbps or 100Mbps Ethernet, but only the high-end models such as 525 and 535 support Gigabit Ethernet. This makes sense when you consider the capacity available on each model: The lower-end models would be overwhelmed by the addition of even a single Gigabit Ethernet interface. As of this writing, the PIX 535 provides 9Gbps of clear-text throughput, the 525 provides 360Mbps, the 515 provides 188Mbps, the 506 provides 20Mbps, and the 501 provides 10Mbps. At the physical layer, the primary issue you will face is to ensure that the correct Ethernet cables are being used and that they are wired correctly. Figure 10.7 shows the pinouts that you should be using for Ethernet and Fast Ethernet cables.

Figure 10.7 Ethernet Cable Pinouts



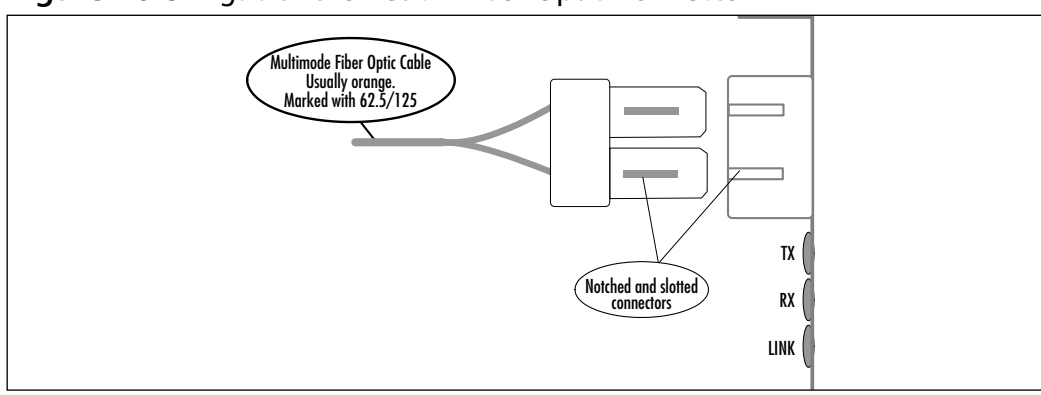
Two wiring schemes for the RJ45 standard are used for 10/100 Ethernet: TA568A and TA568B shown in Figure 10.7. It is important that your cable adhere to one of these standards to prevent interference (crosstalk). If you were to dismantle a RJ45 cable, you would see that there are four pairs of wires. In each pair, the two wires are twisted around each other to minimize crosstalk. If you were to pick wires at random and crimp them into the RJ45 connector to make an Ethernet cable, chances are you would experience problems with your cables. The wiring scheme of the TA568A/B standard is optimized to prevent such interference.

The process of troubleshooting cabling is relatively easy because there are numerous cable testers on the market, ranging from simple pin-checking devices (like the ones you can find at www.copperandfibertools.com/testers.asp) to expensive, full-featured testers such as those offered by Fluke (www.fluke.com). The time that these devices save well justifies their initial cost.

The first step in verifying 10/100 Ethernet copper cables is to visually inspect the cable for breaks. Check the wiring pinouts against Figure 10.7. If they match and appear to be in good physical shape, the next step is to test the cable using a cable tester. Most cable testers will allow you to map the wiring; pin mismatches are a common problem. If you still have problems with after it passes the cable tester, try using a different cable. Chances are, you have a rare bad mix of plastic and metal composition that went into the making of that cable and it is interfering with the cable's ability to transport electrons. If you do not have a cable tester and are not sure of the cable, replace it.

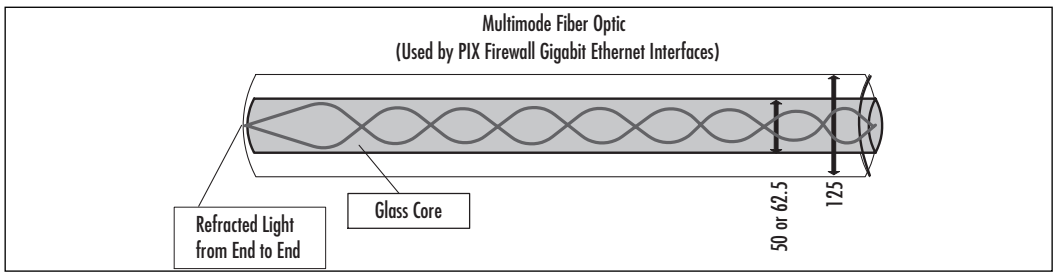
PIX firewall models 525 and 535 support full-duplex Gigabit Ethernet (GE). The GE interfaces use SC multimode fiber optic cables: one strand for receive and the other for transmit, as shown in Figure 10.8. It is important that you cable the wire with the correct cable to the correct connector.

Figure 10.8 Gigabit Ethernet SC Fiber Optic Connector



Fortunately, the SC connector Cisco uses prevents us from inserting the cable incorrectly. The connector on the cable is notched to fit the slotted jack on the interface card. You need to understand a little about fiber optic cables to effectively use them with your PIX firewall. Fiber optic is either single mode or multimode. The PIX firewall GE interfaces use multimode fiber, which refracts light, as shown in Figure 10.9.

Figure 10.9 Multimode Fiber Optic Cable



The fiber optic industry adheres very strictly to its standards. As a result, usually you can visually determine whether you have a multimode or single-mode fiber optic cable attached by its color. Single-mode cables are yellow and have markings down their sides indicating their width in microns. Multimode fiber optic cable used by PIX firewalls is orange and is numerated with either 50 or 62.5 microns, indicating the size of its glass core down which light is sent. The cladding packed in the glass core is the same size for both cables: 125 microns. This is a general rule of thumb only; some manufacturers offer custom colors or do not adhere to the standard color scheme.

As with twisted-pair cable for Ethernet and Fast Ethernet, you can use a cable tester to verify your fiber optic cable. Unlike copper cables, fiber optic cables are very unforgiving of failure to adhere to tight specifications. If you made the cable that you are using and it is not working, odds are very good that you made an error (poor crimping, insufficient polishing, or the like). It is in such situations that the value of a good cable tester becomes apparent. Unless you are a certified fiber optic technician, it is a good idea to leave the fiber optic cable making to the professionals who specialize in it.

Troubleshooting Connectivity

In order to perform its duties, a PIX firewall must be able to reach its destinations. Its ability to pass traffic from source to destination is affected by factors such as routing, address translation, access lists, and so on. Translation can be

particularly critical since all addresses must be translated in order for internal and external networks to communicate with each other.

Get in the habit of executing *clear xlate* to clear any current translations whenever you make a change to NAT, global, static, access lists, conduits, or anything that depends on or is part of translation. Since translation is mandatory on PIX firewalls, this covers just about any feature you can configure. Failure to delete existing translations will cause unexpected behavior.

Remember how interfaces of different security levels work with each other. Traffic from a higher security level to a lower security level is permitted by default but still requires translations to be set up. Traffic from a lower security level to a higher security level (such as outside to inside) requires an access list or conduit, as well as corresponding translations.

We covered syslog extensively in Chapter 6, but it bears repeating that you should get in the habit of checking log messages. Syslog provides an ongoing, real-time report of activities and errors—information that can be vital to troubleshooting success. The information syslog provides can help you take your first or next step, so ensure that you develop your syslog reading habits. This can be particularly useful in identifying errors with access lists and translation. For example, if a host on a lower security level interface wants to communicate with a host on a higher security level interface and translation is enabled for it, but no conduit or access list is configured, the following message will be logged:

```
106001: Inbound TCP connection denied from x.x.x.x/x to x.x.x.x/x
```

This is your first clue that you need an access list or conduit to permit this access. If the reverse is the case (access list or conduit is present, but no translation is configured), the following message will be logged:

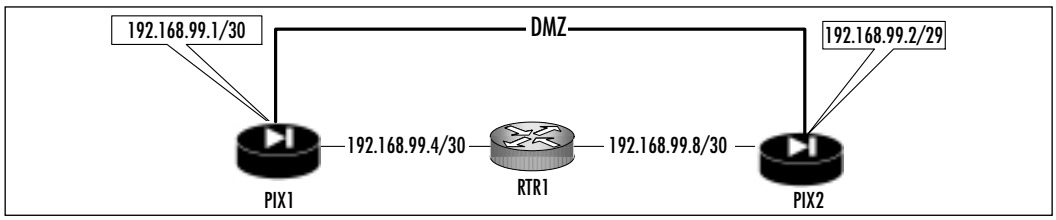
```
305005: No translation group found for...
```

For more information about syslog message numbers and descriptions, see www.cisco.com/univercd/cc/td/doc/product/iaabu/pix/pix_61/syslog/pixemsgs.htm.

Checking Addressing

As with any IP device, unless basic IP addressing and operation are configured correctly and working, none of your PIX firewall troubleshooting efforts regarding routing, access lists, and translation will matter. This point cannot be overstressed: Addressing must be correct in order for the PIX firewall to function. Figure 10.10 shows PIX1 and PIX2 connected to each other.

Figure 10.10 IP Addressing Problem



In the figure, there is an addressing problem on the LAN connecting the two firewalls (which is labeled DMZ in the configuration). For starters, PIX1 has a subnet mask of /30, while FW2 has a mask of /29 for the DMZ network (192.168.99.0), a common network between them. This is confirmed using the *show ip address* command on both firewalls. Notice the differences highlighted in the command output shown in Figure 10.11.

Figure 10.11 IP Address Configuration

```
PIX1# show ip address
System IP Addresses:
         ip address outside 192.168.99.5 255.255.255.252
         ip address DMZ 192.168.99.1 255.255.255.252
Current IP Addresses:
         ip address outside 192.168.99.5 255.255.255.252
         ip address DMZ 192.168.99.1 255.255.255.252

PIX2# show ip address
System IP Addresses:
         ip address outside 192.168.99.9 255.255.255.252
         ip address DMZ 192.168.99.2 255.255.255.248
Current IP Addresses:
         ip address outside 192.168.99.9 255.255.255.252
         ip address DMZ 192.168.99.2 255.255.255.248
```

The fix here is simply to correct the mask on PIX2. As on Cisco routers, the *show interface* command can also be used to check addressing on your PIX firewall, as shown in Figure 10.12.

Figure 10.12 Address Verification Using the show interface Command

```
PIX1# show interface
interface ethernet0 "DMZ" is up, line protocol is up
 Hardware is i82559 ethernet, address is 0008.e317.ba6b
 IP address 192.168.99.1, subnet mask 255.255.255.252
 MTU 1500 bytes, BW 100000 Kbit half duplex
        2 packets input, 258 bytes, 0 no buffer
        Received 0 broadcasts, 0 runts, 0 giants
        0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
        11 packets output, 170 bytes, 0 underruns, 0 unicast rpf drops
        0 output errors, 0 collisions, 0 interface resets
        0 babbles, 0 late collisions, 0 deferred
        0 lost carrier, 0 no carrier
        input queue (curr/max blocks): hardware (128/128) software (0/1)
        output queue (curr/max blocks): hardware (0/2) software (0/1)
```

Regardless of the method you use, verify that all interface IP addresses are correct before proceeding any further in your troubleshooting efforts. Incorrect addressing will prevent advanced features of the PIX firewall from working, even if you configure them correctly. After all, all traffic must pass through at least two interfaces, and the interfaces must be addressed correctly.

Checking Routing

The inability to reach a destination is a prime indicator of routing problems. Such problems can be complex to troubleshoot, but using a structured approach to isolate the cause can ease troubleshooting. The PIX firewall uses both static and dynamic routing. For dynamic routing, the PIX supports only RIP as a routing protocol; otherwise, the routing information it has is manually entered in the form of static routes. We open our routing verification discussion with a review of the various routing options available on the PIX firewall and how they interact.

NOTE

The only routing protocol supported by the PIX firewall at this writing is RIP (version 1 and version 2). RIP is discussed briefly in this chapter as it pertains to the PIX firewall.

First, let's review the techniques you use to configure routing on your PIX, starting with the simplest (default route) and onward to using RIP to learn routes. In the simplest configuration, the PIX firewall is configured only with a static default route. For example:

```
route outside 0.0.0.0 0.0.0.0 192.168.99.2 metric 1
```

This command states that all traffic that does not match any of the local interfaces will be sent to the next hop of 192.168.99.2. Assuming this is the only static route configured on the firewall in Figure 10.13, all traffic destined for a non-local interface on the PIX firewall will be forwarded to RTR1 to reach its final destination. A single static route such as this one works well for the simple configuration in Figure 10.13, but what happens if we have a more complex architecture, such as the one shown in Figure 10.14?

Figure 10.13 Default Route Example

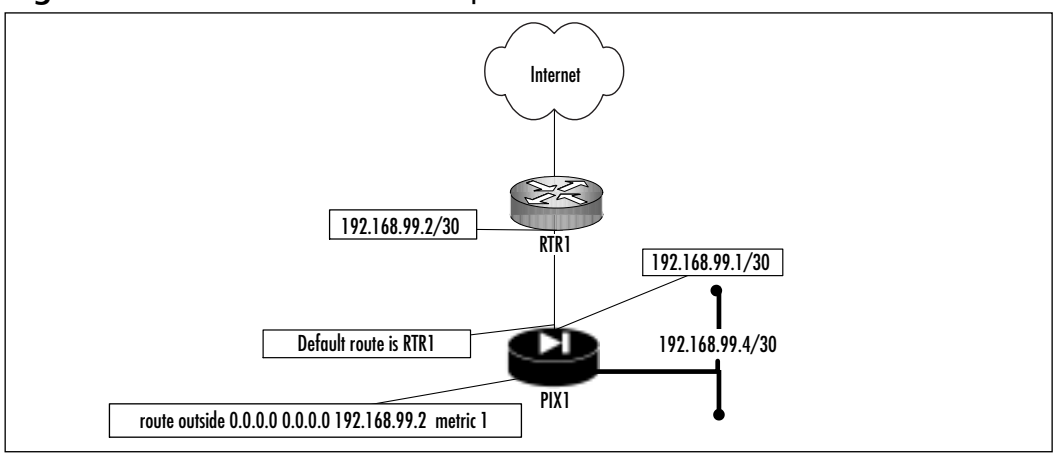
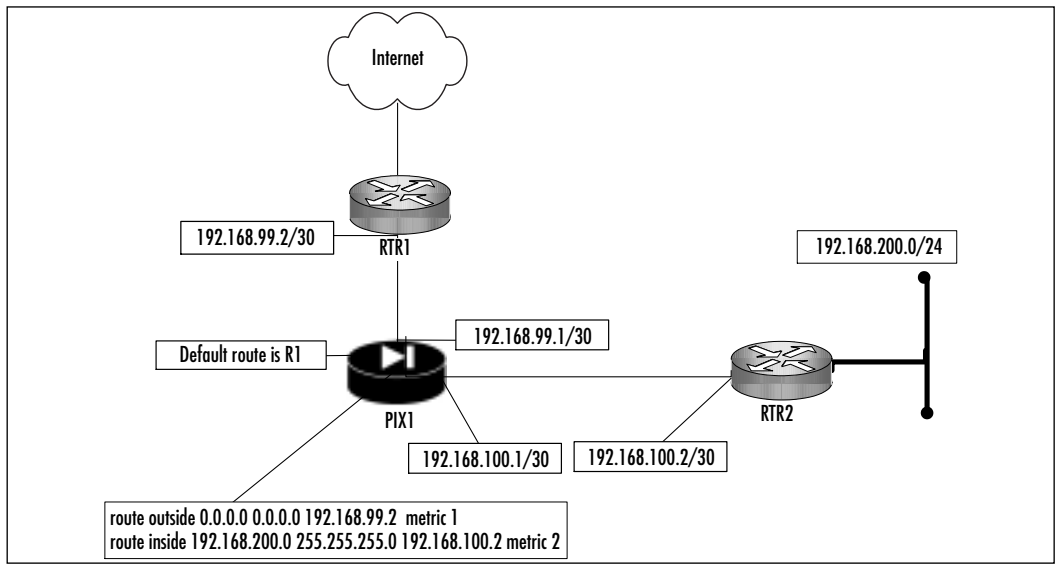


Figure 10.14 shows that the traffic from PIX1 must be forwarded to R2 to reach 192.168.200.0/24. If we used only a default route, any traffic for 192.168.200.0/24 would be sent to RTR1 and would never reach its destination. We can resolve this issue by adding a static route on PIX1 so it knows where to forward traffic destined to 192.168.200.0/24. This is accomplished by adding another (more specific) route to the PIX1 configuration:

```
route inside 192.168.200.0 255.255.255.0 192.168.100.2 metric 2
```

Figure 10.14 Static Routes



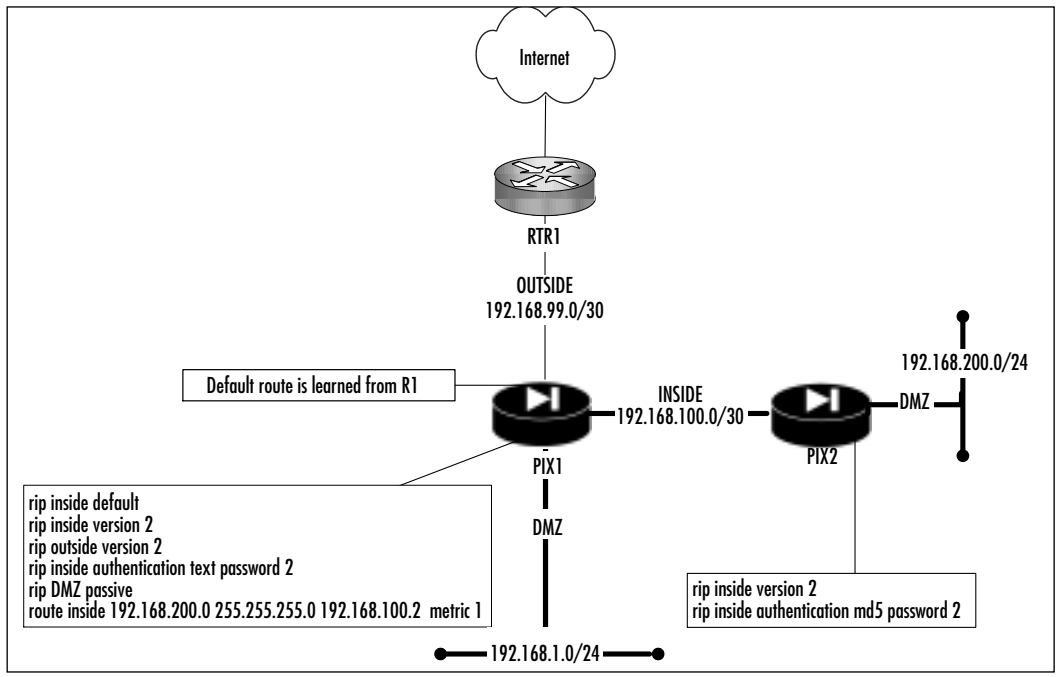
In addition to using these static methods for routing, the PIX firewall supports dynamic routing using RIP version 1 or version 2. Unlike the wide range of options available for RIP on Cisco routers, the RIP commands on the PIX firewall are very sparse.

```
[no] rip <if_name> default
[no] rip <if_name> passive
[no] rip <if_name> version {1 | 2}
[no] rip <if_name> authentication [text | md5] key <key_id>
```

We will not spend an inordinate amount of time debating the merits of RIP as a routing protocol. Suffice to say, the *default* keyword means that the PIX firewall advertises a default route out that interface. The *passive* keyword configures RIP to listen on, but not advertise out, a particular interface. The *version* keyword is used to set the version of RIP that the PIX firewall will use. RIP peers can authenticate each other to ensure that they send and receive updates from legitimate peers. RIP is enabled on a per-interface basis.

In Figure 10.15, we have replaced our statically routed network with RIP version 2. Notice how this replacement has changed the routing picture, enabling the PIX firewall to better adapt to network changes.

Figure 10.15 RIP Routing



On PIX firewalls, RIP does not advertise from interface to interface. In Figure 10.15, PIX1 is listening for updates on its DMZ network and is learning any routes that might be present behind that network. As a result, PIX1 will know how to reach those networks. Since the *passive* keyword is used, PIX1 will not advertise any RIP routes out its DMZ interface. However, PIX1 will *not* advertise those routes to PIX2 or RTR1. This is a limitation of RIP in the PIX firewall that needs to be resolved by adding a default route to PIX2 (which our configuration has) and a static route on R1 to reach any networks behind PIX1's DMZ interface. What PIX1 will advertise is any of its directly connected interfaces and default routes, so R1 and PIX2 will be able to reach any directly connected network on PIX1. PIX2 will be able to reach the networks behind PIX1's DMZ interface since PIX1 is the default route for PIX2.

This limitation of RIP might not be such a limitation. In actual practice, any addresses that leave or enter PIX1 related to the outside interface would actually be translated. In the case of RTR1, it does not need to know about the networks behind PIX1's DMZ network since those addresses would be translated to a public address, which RTR1 would know to send to PIX1 for processing.

One problem is quite apparent in our configuration in Figure 10.15. There is an authentication mismatch between PIX1 and PIX2. PIX1 is using a clear text password for authentication, while PIX2 is using MD5. Although the password is the same on both sides, the encryption technique is different. The result is that RIP routing will not work between them, as disagreement on the password encryption technique will prevent the peers from authenticating to each other, which will prevent the exchange and acceptance of routing updates.

Another potential showstopper that you need to be alert for is conflicting versions of RIP. The most significant difference is that RIP version 1 broadcasts to an all-hosts broadcast address of 255.255.255.255. Version 2 generally multicasts to the reserved IP multicast address of 224.0.0.9. Additionally, version 2 supports authentication, whereas version 1 does not. When troubleshooting routing problems with RIP, look at the configuration of the devices where routing is not working, and check to make sure that all your routing peers agree on the version. If you are using RIP version 2 with authentication, ensure that the same password and the same encryption method are used on both. Support for RIP version 2 was introduced in PIX software version 5.1. Prior versions cannot interoperate with RIP version 2 speakers, so keep the RIP version differences in your mind as you troubleshoot. Support for RIP version 2 multicast was introduced in version 5.3. Prior versions could only handle broadcasts.

Having reviewed how the PIX gets its routes, we now turn our attention to troubleshooting when the PIX is unable to reach a particular destination or when it does not have a route to a particular destination. Your tools of choice for troubleshooting routing issues on the PIX are primarily *show route, show rip,* and *ping* . Determine if there is a reachability problem by attempting to ping the destination. If that fails, use *show route* to determine if there is a route (static or RIP) to reach the network. You can use the *show rip* command to confirm your dynamic routing configuration. The *ping* command should be a litmus test to verify that the destination cannot be reached. The syntax of the *ping* command is as follows:

```
ping [<if_name>] <ip_address>
```

For example:

```
PIX1# ping 192.168.99.2
          192.168.99.2 response received – 20ms
          192.168.99.2 response received – 20ms
          192.168.99.2 response received – 20ms
```

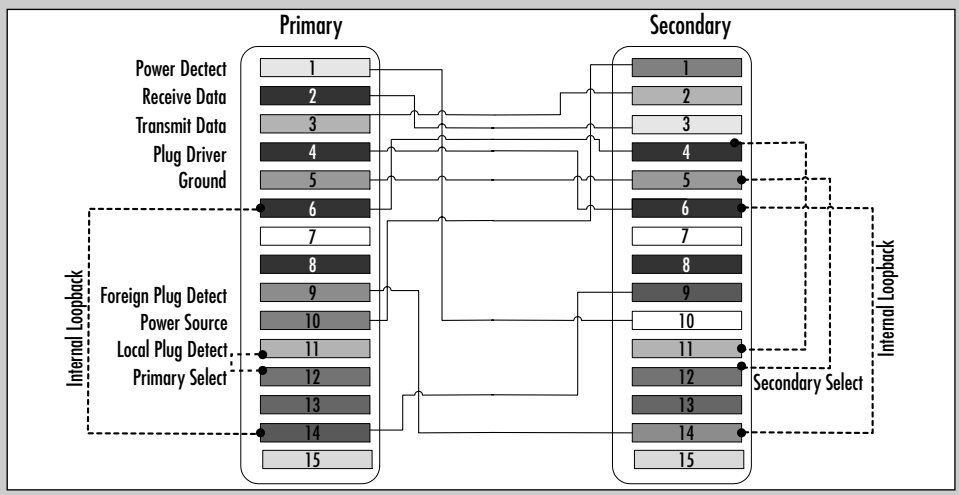
Configuring & Implementing…

Failover Cable

Cisco provides a wonderful feature called *failover*, wherein the configuration and operations of one firewall are mirrored to a backup firewall. Failover is discussed in detail in Chapter 8. When using standard failover with the failover cable, it is the cable that determines which firewall is the primary and which is the secondary unit in a pairing. The cable makes this determination based on which end is plugged into which firewall.

As part of your PIX firewall troubleshooting knowledge, you need to know the pinout scheme used by this cable. To that end, we have provided a detailed schematic in Figure 10.16. If failover is not working, you need to know what your cable configuration should look like when you analyze it with a cable tester.

Figure 10.16 Failover Cable Pinout



Although all the wires in the DB15 connector at each end are important, you can see that certain wires are cross-connected at each end to distinguish the primary end from the secondary end. The primary firewall is configured by cross-connecting wire 11 (local plug detect) to wire 12 (primary select). The secondary firewall is determined by cross-connecting wire 12 (secondary select) to wire 5 (ground). Knowing the wiring scheme can enable you to not only to check your failover cable but to also build one from scratch if necessary.

Does the PIX have a default route, a static route, or even a dynamically learned route? Check your routing table with the *show route* command. For example:

```
PIX1# show route
      outside 192.168.99.0 255.255.255.252 192.168.99.1 1 CONNECT static
      inside 192.168.100.0 255.255.255.252 192.168.100.1 1 CONNECT static
      DMZ 192.168.1.0 255.255.255.0 192.168.1.1 1 CONNECT static
```

In our case, 192.168.99.2 is on our directly connected outside network. To perform a side-by-side comparison of RIP peers, use the *show rip* command. In Figure 10.17, we are looking at the RIP configuration of PIX1 and PIX2; notice how the mismatches between the versions and authentication technique are readily apparent.

Figure 10.17 Identifying RIP Configuration Errors

```
PIX1# show rip
rip inside default
rip inside version 1
rip outside version 2
rip inside authentication text cisco1 2
rip DMZ passive

PIX2# show rip
rip inside version 1
rip outside version 1
rip inside authentication md5 cisco2 2
rip DMZ passive
```

The result of this configuration is that RIP will *not* work between PIX1 and PIX2 since they do not agree on any of the parameters. A corrected configuration that will work is provided in Figure 10.18.

Figure 10.18 RIP Configuration Fixed

```
PIX1# show rip
rip inside default
rip inside version 2
rip outside version 2
```

Continued

Figure 10.18 Continued

```
rip inside authentication md5 cisco2 2
rip DMZ passive

PIX2# show rip
rip inside version 2
rip outside version 2
rip inside authentication md5 cisco2 2
rip DMZ passive
```

We conclude our discussion of RIP with the *clear rip* command, which should only be used when you have made a definite decision that you no longer need to use RIP. This command removes all existing RIP commands and parameters from the configuration.

Checking Translation

The PIX firewall performs address translation. In order for internal networks to communicate with external networks, and vice versa, addresses must be translated. Translation is *not* optional. Recall from Chapter 3 that translation is the act of translating one IP address to another, which can be configured as one to one (NAT) or many to one (PAT).

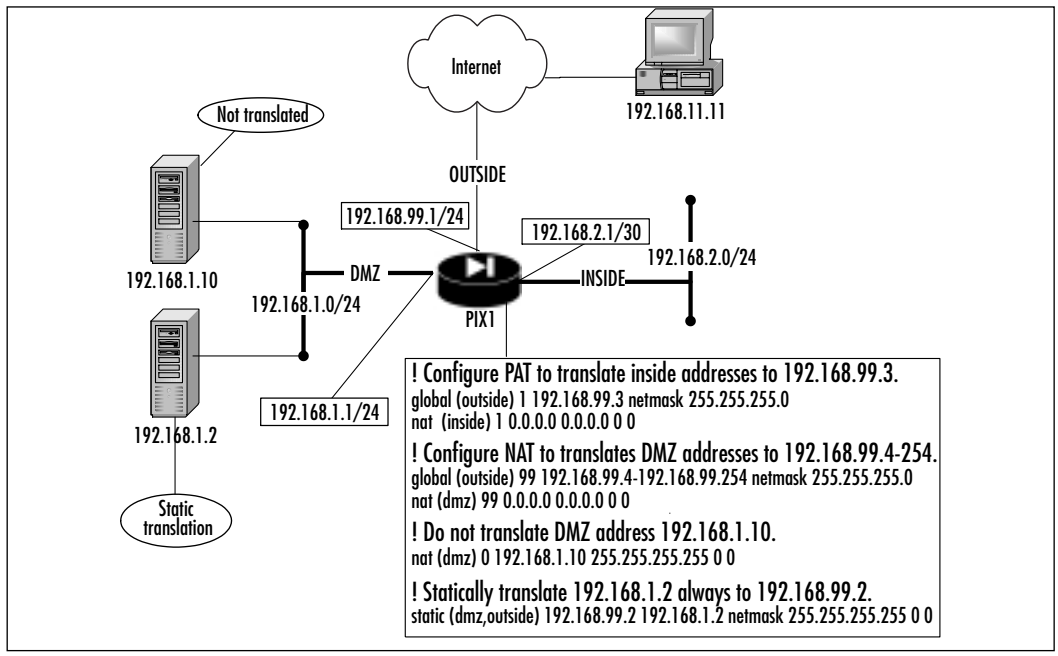
NOTE

To pass traffic through the PIX traffic, you must translate it, even if this means you will translate IP addresses to themselves.

We discussed translation at some length in Chapter 3. In this chapter, we quickly review some key concepts using Figure 10.19, which shows all the possible translation scenarios that you can have on your PIX firewall.

Figure 10.19 shows a PIX firewall, PIX1, connected to three networks: inside, DMZ, and outside. The addresses on the inside network are serviced using PAT. The DMZ has two hosts on it: one that is not translated (in reality, it is just translated to itself) and one that is statically translated. All remaining addresses on the DMZ are dynamically translated using a range of IP addresses associated with the outside network.

Figure 10.19 Translation in Action



In the PIX world, translation is necessary to provide connectivity. When translation does not work, you need to know where to start and finish your troubleshooting. Cisco provides several commands that you can use to validate various aspects of translation. We start with a review of the various translation configuration commands and how to effectively institute them. Let's review the configuration in Figure 10.19.

First, look at which private addresses are being translated to which public addresses. This information will determine if the translation parameters have been configured correctly. Two commands used to perform this task are *show nat* and *show global*:

```
PIX1# show nat
nat (dmz) 0 192.168.1.10 255.255.255.255 0 0
nat (inside) 1 0.0.0.0 0.0.0.0 0 0
nat (dmz) 99 0.0.0.0 0.0.0.0 0 0
PIX1# show global
global (outside) 99 192.168.99.4-192.168.99.254 netmask 255.255.255.0
global (outside) 1 192.168.99.3 netmask 255.255.255.0
```

Our NAT configuration specifies a nontranslation for the DMZ server at address 192.168.1.10 network (as evidenced by the *nat 0* command). The *nat 99* specifies that all remaining addresses in the DMZ should be translated. The global command defines two pools of addresses to be used for translation purposes. The numerical ID is referenced by the NAT command to perform the actual translation. The *global 99* command is used for NAT, whereas *global 1* with its single IP address is used for PAT. In actual practice, you would know at this point if you had configured the translation parameters correctly. Both of these commands provide enough data for you to make this determination. Once you have corrected any errors (the most common being typos or incorrect IP addresses), you can then check to see if connections are being made and translated. The next step is to determine if connections have been made by using the *show conn detail* command:

```
PIX1# show conn detail
1 in use, 1 most used
Flags: A - awaiting inside ACK to SYN, a - awaiting outside ACK to SYN,
       B - initial SYN from outside, D - DNS, d - dump,
       E - outside back connection, f - inside FIN, F - outside FIN,
       G - group, H - H.323, I - inbound data, M - SMTP data,
       O - outbound data, P - inside back connection,
       q - SQL*Net data, R - outside acknowledged FIN,
       R - UDP RPC, r - inside acknowledged FIN, S - awaiting inside SYN,
       s - awaiting outside SYN, U - up
TCP outside:192.168.11.11/24 dmz:192.168.99.2/80 flags UIO
```

The workstation has established a connection to our HTTP server on the DMZ network (as confirmed by its destination port, 80). Notice that the workstation established the connection to the public address of this server rather than to its internal DMZ address (192.168.1.2), which it cannot reach. Now we have a valid connection attempt, but has the translation taken place as it should? To determine that, we must use the next command in our toolbox, *show xlate detail*:

```
PIX1# show xlate detail
1 in use, 1 most used
Flags: D - DNS, d - dump, I - identity, i - inside, n - no random,
       o - outside, r - portmap, s - static
TCP NAT from DMZ:192.168.1.2/80 to outside:192.168.99.2/80 flags ri
```

This command displays a current listing of active translation slots. The output of this command confirms that our host's attempt to access the Web server at

192.168.99.2 has resulted in the correct translation to 192.168.99.2. Such verification is particularly important if you are providing services that must be accessible by outside users.

There is one more command that we can use to gather information about our translation operations. It is a *debug* command and, as such, should be used sparingly to conserve firewall resources. This command can serve two functions: tracking and decoding packet-level activity between hosts (such as the traffic between our workstation and the Web server) or it can be used if you need to determine exactly which addresses need to be translated and granted access. The latter part of this statement needs to be explained more fully. Assuming that we did not know exactly what the source address of our workstation was going to be, it would be helpful to capture information on its attempts to connect to the DMZ Web server. The command that can provide us with the copious information we need is the *debug packet* command. The syntax of the command is as follows:

```
debug packet <if_name> [src <source_ip> [netmask <mask>]] [dst <dest_ip>
    [netmask <mask>]] [[proto icmp] | [proto tcp [sport <src_port>]
        [dport <dest_port>]] | [[proto udp [sport <src_port>] [dport
            <dest_port>]] [rx | tx | both]
```

In our case, the command we would actually enter to find out which addresses are attempting to use our Web server is:

```
PIX1(config)# debug packet outside src 0.0.0.0 netmask 0.0.0.0 dst 192
    .168.99.2 netmask 255.255.255.0 rx
```

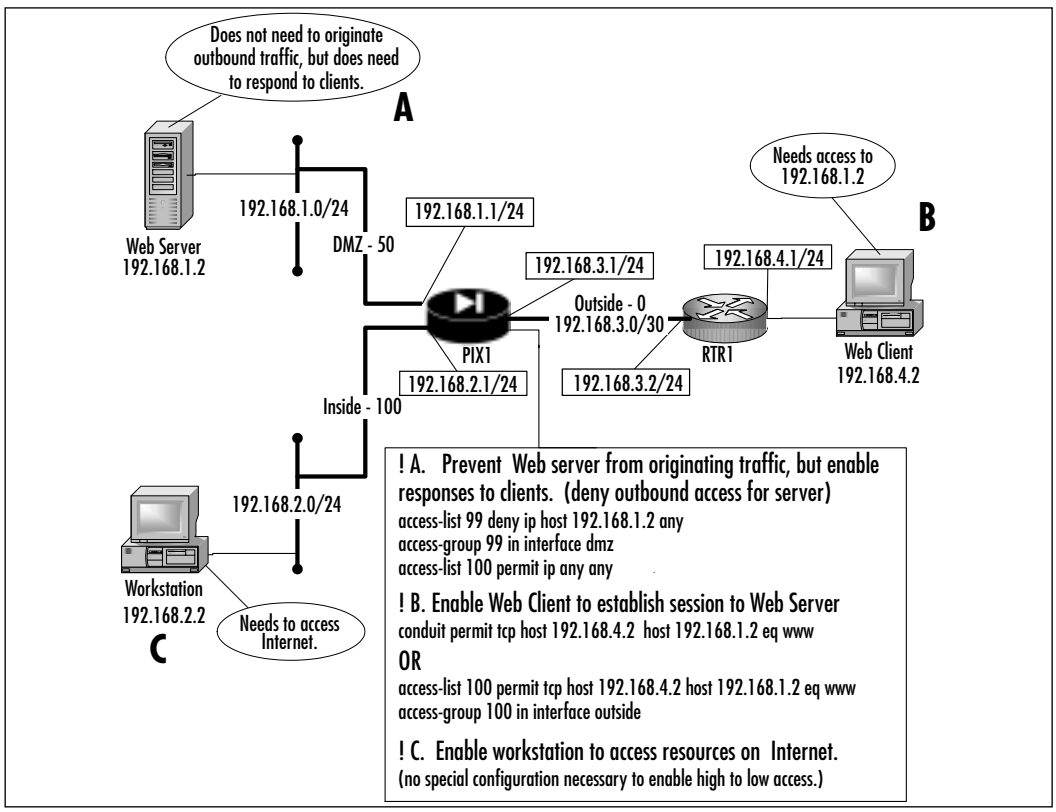
This command captures packet data that comes into the outside interface destined for the Web server's public IP address. Since we do not know exactly which protocols (TCP, UDP, or ICMP) will be used, we have opted not to specify one. After we have captured our data, we can then determine which translation parameters we need to enter.

Checking Access

The PIX firewall provides several mechanisms for controlling access through it. In this section, we cover several of these mechanisms and discuss some ways to monitor and verify their functionality. The default state of the PIX firewall is to permit access to sessions originated from a higher security-level interface to a lower security-level interface, as long as a translation is configured. Traffic that originates from a low security-level interface to a high security-level interface has to be specifically permitted using conduits or access lists (and of course, translations).

The *conduit* command is a special form of an access list. It is used to permit traffic from a lower security-level interface to a higher security-level interface. Figure 10.20 shows several common access scenarios with various hosts needing access to each other. The Web client (security level 0) will be accessing the Web server (security level 50); the default behavior of the PIX firewall is to forbid such traffic. The workstation (security level 100) needs to access Internet resources using the outside network. The figure also provides the configuration necessary to enable the access needed by the various hosts and servers, which are denoted *A, B,* and *C* for ease of discussion. The assumption is that all translation parameters have been configured and are working correctly, which enables us to focus on specific access issues. The addresses shown are used for discussion, but in your mind, assume that they have been translated.

Figure 10.20 Access Scenario



The Web server needs to be prevented from originating sessions to networks located off the DMZ network but must be able to respond to service requests from the Web client located on the outside network. To accomplish this goal, we

created an access list to deny 192.168.1.2 from accessing anything and applied it to the DMZ interface. Then we created a conduit to permit 192.168.4.2 to access Web services (TCP port 80) on 192.168.1.2. Alternatively, we could have used an access list to accomplish the same thing, as shown in Figure 10.20. The option to use access lists instead of conduits is available only on PIX firewall software versions 5.1 and later. It is important to note that Cisco recommends that you avoid mixing access lists and conduits. Additionally, access lists take precedence over conduits. In the PIX environment, access lists have one and only one direction: in. The *access-group* command applies the access list to traffic coming into the designated interface.

The inside workstation (denoted by *C*) needs to be able to access resources on the Internet. The inside interface has a security level of 100, the highest possible security level. Recall that hosts on higher security-level interfaces can access hosts on lower security-level interfaces without any special configuration to permit responses to return. This is exactly the case with this workstation, so we need no special configuration.

Problems with lack of access become apparent when machines are unreachable. Since access control mechanisms such as access lists and conduits have a close interdependent relationship with translation, you should validate the translation configuration first. Once that is confirmed, begin your access troubleshooting. Access problems can include typos, overly restrictive or loose access lists or conduits, the wrong networks being denied or permitted access, or access lists applied to the wrong interface. Here we demonstrate several commands that you can use to verify access.

Recall that a conduit is a hole in your firewall security that permits hosts on a lower security level access to resources on a higher security level. The main command for verifying conduit configuration is *show conduit*. For example:

```
PIX1# show conduit
conduit permit tcp host 192.168.4.2 host 192.168.1.2 eq www (hitcnt=3)
```

This conduit permits 192.168.4.2 to access the Web server at 192.168.1.2. This is the only PIX command for checking conduits. With the option provided in version 5.1 to use access lists instead, conduits are gradually being phased out in favor of the more standard access lists. When that happens, you can remove all conduit parameters from your PIX firewall configuration using the *clear conduit* command. This is a slightly schizophrenic command, depending on where it is it used. If used at the privileged command prompt as *clear conduit counters*, it "zeroizes" the hit counter. If *clear conduit* is used in the Configuration mode, it removes all conduit statements from the PIX firewall configuration.

Access lists, another access control mechanism, offer more troubleshooting tools than conduits do. The *show access-list* command can be used to confirm which access lists are configured on the PIX firewall and what they are permitting and denying:

```
PIX1# show access-list
access-list 99; 2 elements
access-list 99 deny ip host 192.168.1.2 any (hitcnt=1)
access-list 99 permit ip any any (hitcnt=0)
access-list 100 permit tcp host 192.168.4.2 host 192.168.1.2
eq www
     (hitcnt=5)
```

This command was executed on the firewall in Figure 10.20. Recall that an access list only affects incoming traffic to an interface. Once you have confirmed that the access list is configured as it should be, the next troubleshooting step is to verify that it has been applied to the correct interface. Cisco provides the *show access-group* command for this purpose. For example:

```
PIX1# show access-group
access-group 99 in interface dmz
access-group 100 in interface outside
```

The *in* keyword is mandatory and serves as a reminder that the access list is applied *only* to traffic coming into the interface. Cisco provides a *debug* command for troubleshooting access list events as they occur. Be aware that when you use this command, it debugs all access lists. There is no option to do real-time monitoring of a particular access list. This can generate copious amounts of data, especially if you execute it on a high-traffic PIX firewall. As with any *debug* command, use it sparingly and only if you know what you are searching for. The *debug access-list* command can provide feedback on your access list and whether it is permitting or denying the traffic that it should. The command syntax is as follows:

```
debug access-list {all | standard | turbo}
```

Another access control mechanism is *outbound/apply,* but Cisco recommends that it not be used. Cisco recommends that you use the access list features of the PIX firewall instead. The *outbound/apply* commands were the precursor to the access list feature and are still available and supported by the PIX firewall software. However, these commands suffer from a very awkward syntax, are fairly limited, and can be frustrating to troubleshoot. The *outbound* command was designed to control access of inside users to outside resources. Having said all

that, a working familiarity with the command is handy for when you encounter situations in which it is still used. The syntax for the *outbound* command is as follows:

```
outbound <ID> {permit | deny | except} <ip_address> [<netmask>] [<port>
    [-<port>]] [tcp | udp| icmp]
```

The *ID* parameter specifies a unique identifier for the outbound list. You can either configure a *permit* rule, a *deny* rule, or an *except* rule (which creates an exception to a previous outbound command). Unlike access lists, outbound lists are not processed from top to bottom. Each line is parsed regardless of whether there is a match or not. Cisco recommends that all outbound lists start with a deny all (*deny 0 0 0*), followed by specific statements allowing access. The net effect is cumulative. How the PIX firewall uses the outbound list depends on the syntax of the *apply* command:

```
apply [<interface>] <OUTBOUND_LIST_ID> {outgoing_src | outgoing_dest}
```

When the *outgoing_src* parameter is used, the source IP address, destination port, and protocol are filtered. When the *outgoing_dst* parameter is used, the destination IP address, port, and protocol are filtered. It is vital you understand that the outbound list does not determine whether the IP address it uses is either a source or a destination; the *apply* command does that. This can be a major troubleshooting headache because an outbound list could be configured correctly but might not work because the *apply* command is configured incorrectly. When troubleshooting *outbound*, ensure that you check the *apply* configuration as well. When multiple rules match the same packet, the rule with the best match is used. The best-match rule is based on the netmask and port range. The stricter the IP address and the smaller the port range, the better a match it is. If there is a tie, a *permit* option takes precedence over a *deny* option.

Here is an example of *outbound/apply*:

```
PIX1(config)# outbound 99 deny 0 0 0
PIX1(config)# outbound 99 permit 0.0.0.0 0.0.0.0 1-1024 tcp
PIX1(config)# outbound 99 except 192.168.2.0 255.255.255.0
PIX1(config)# apply (inside) 99 outgoing_src
```

In this example, the first statement denies all traffic, the second line permits any host access to TCP ports 1-1024 on any host, and the third line denies the 192.168.2.0/24 network from access to any TCP ports permitted by the second line. We are using the *outgoing_src* keyword, meaning that the IP addresses referenced are source addresses.

Cisco only provides a few commands for checking *outbound/apply* parameters. First, do not forget to do a *clear xlate* after configuring *outbound/apply*. Use *show outbound* to view the outbound lists that are configured. The *show apply* command identifies the interfaces and direction to which the outbound lists have been applied. No *debug* commands are associated with *outbound/apply*. Given that access lists have now superseded *outbound/apply*, you would be better served in terms of both configuration and support to use them instead. Not only do access lists conform to the standard Cisco syntax, they also offer better and easier-to-understand filtering.

One feature does not seem to be access related, but since it curtails the operations of selected protocols, one can argue that access to certain features of the "protected" protocol have been negated. As discussed in Chapter 4, the PIX firewall software provides application inspection features through the *fixup* command. There is a standard set of protocols for which the *fixup* capability is enabled automatically, such as HTTP, SMTP, FTP, and so on. This protocol sometimes disables certain commands or features in the target protocols to prevent malicious misuse. To determine for which protocols *fixup* is enabled, run the *show fixup* command. For example:

```
PIX1# show fixup
fixup protocol ftp 21
fixup protocol http 80
fixup protocol h323 h225 1720
fixup protocol h323 ras 1718-1719
fixup protocol ils 389
fixup protocol rsh 514
fixup protocol rtsp 554
fixup protocol smtp 25
fixup protocol sqlnet 1521
fixup protocol sip 5060
fixup protocol skinny 2000
```

Troubleshooting IPsec

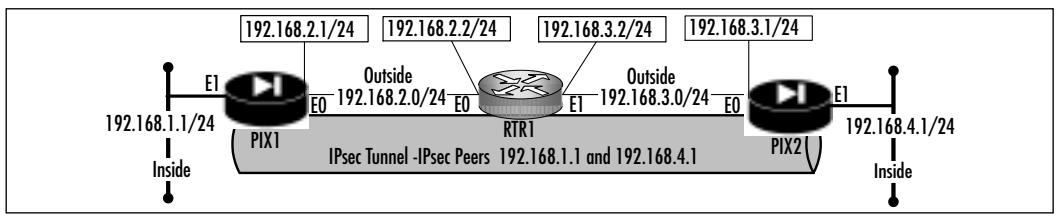
Recall from Chapter 7 that IPsec is used on the PIX firewall for the establishment of a secure VPN tunnel between two endpoints for the purpose of securely exchanging data over IP. IPsec can be configured using IKE with RSA key exchange, IKE with CA certificates, IKE with preshared keys, or using preshared keys sans IKE (called *manual IPsec*). When using manual key exchange, you simply

create a shared secret that is the same on both endpoints; this technique is not only a security risk, but it has scalability issues.

We will not rehash the configuration steps necessary to deploy IPsec on PIX firewalls, because that topic was covered in Chapter 7. We instead focus our efforts on using the tools Cisco provides to troubleshoot IPsec problems using an IPsec with IKE preshared key configuration. Misconfigurations, mismatched parameters, keys, routing, IP addressing issues, and other problems can conspire to make IPsec fail. You need to be able to isolate and resolve these issues by first recognizing the symptoms and then using the correct tools to pinpoint the cause.

Figure 10.21 shows a simple point-to-point IPsec tunnel configured between PIX1 and PIX2. IPsec is a complicated technology and very unforgiving of errors. A single error can prevent your IPsec configuration from working at all. Therefore, you will find that the bulk of your labors will be focused on setting IPsec correctly in the first place.

Figure 10.21 IPsec Configuration



Here we introduce several commands and procedures that you can use to check your configuration.

```
! PIX1 Configuration snippets
nat 99 0.0.0.0 0.0.0.0
global (outside) 99  192.168.2.10-192.168.2.254 netmask 255.255.255.0
route outside 0.0.0.0 0.0.0.0 192.168.2.2
static (inside, outside) 192.168.2.10 192.168.1.1 netmask 255.255.255.255
conduit permit ip 192.168.3.0 255.255.255.0 any
isakmp enable outside
isakmp policy 99 authen  pre-share
isakmp policy 99 encryption des
isakmp policy 99 group 1
isakmp policy 99 hash md5
isakmp policy 99 lifetime 9999
isakmp identity address
isakmp key cisco address 192.168.3.1
```

```
access-list 99 permit ip 192.168.0.0 255.255.252.0 any
crypto ipsec transform-set FW1 ah-md5-hmac esp-des esp-md5-hmac
crypto map FW1 1 ipsec-isakmp
crypto map FW1 2 set peer 192.168.3.1
crypto map FW1 3 match address 99
crypto map FW1 2 set peer 192.168.3.1
crypto map FW1 interface outside

! PIX2 Configuration snippets
nat 99 0.0.0.0 0.0.0.0
global (outside) 99  192.168.3.10-192.168.2.254 netmask 255.255.255.0
route outside 0.0.0.0 0.0.0.0 192.168.3.2
static (inside, outside) 192.168.3.10 192.168.4.1 netmask 255.255.255.255
conduit permit ip 192.168.3.0 255.255.255.0 any
isakmp enable outside
isakmp policy 99 authen  pre-share
isakmp policy 99 encryption des
isakmp policy 99 group 1
isakmp policy 99 hash md5
isakmp policy 99 lifetime 9999
isakmp identity address
isakmp key cisco address 192.168.2.1
access-list 99 permit ip 192.168.0.0 255.255.252.0 any
crypto ipsec transform-set FW1 ah-md5-hmac esp-des esp-md5-hmac
crypto map FW1 1 ipsec-isakmp
crypto map FW1 2 set peer 192.168.2.1
crypto map FW1 3 match address 99
crypto map FW1 interface outside
```

There are several issues with this configuration. For starters, the IPsec peering between PIX1 and PIX2 is to their inside addresses rather than their outside addresses. Although this might work, Cisco does not recommend it as a method to deploy IPsec. Additionally, the addresses for the peering have been statically translated to an outside address. This presents a problem in that the actual source address of IPsec traffic will not match when it reaches the distant end, and the hash values will also be incorrect. Solving this problem involves disabling translation for the addresses used for establish peering (*nat 0*), adding a route to the internal addresses on each firewall, and permitting the addresses to enter the firewall.

IKE

The chief mission of IKE is to negotiate parameters for IPsec by establishing a secure channel over which IPsec will establish its peering. In other words, IKE does the necessary preconfiguration by establishing the security associations to protect IPsec during its negotiations and operations.

IKE peers create the necessary security association if they both agree on a common security policy, which includes using the same encryption, authentication, Diffie-Hellman settings, and hash parameters. Without this agreement, IKE peering will not take place, and IPsec peering will be unable to proceed. IKE authenticates IPsec peers, determines the encryption methods that will be used, and negotiates the various parameters to be used by IPsec, such as encryption, authentication, and keys. In order for IPsec to proceed, IKE must be configured perfectly and working.

Recall from Chapter 7 that IKE works in two phases. In Phase I (main mode), it establishes the security association necessary for two firewalls to become IKE peers. This includes the exchange and search for common security policies until both peers come to an agreement. During Phase II (quick mode), IKE establishes the security association necessary to protect IPsec during its negotiations and operations. Once Phase II is complete, IPsec can then complete its peering.

Before deploying IKE on your PIX firewall, ensure that each peer can reach the IP address of the other side. If an underlying hardware, network, or translation issue prevents the peers from reaching each other, fix it using the structured methodology presented earlier in this chapter. You can verify reachability using ping.

Cisco provides several commands that you can use to check your IKE configuration and operation; let's look at those commands. The *show isakmp* command shows how IKE is configured on the PIX firewall. For example:

```
PIX1# show isakmp
isakmp enable outside
isakmp key ******** address 192.168.3.1 netmask 255.255.255.255
isakmp identity address
isakmp policy 99 authentication pre-share
isakmp policy 99 encryption des
isakmp policy 99 hash md5
isakmp policy 99 group 1
isakmp policy 99 lifetime 9999
```

The *show isakmp* or *show crypto isakmp* commands display the current IKE parameters configured on a PIX firewall. Notice how the key is hidden to protect its security. You should run this command on both peers and compare the resulting output to ensure that there will be agreement on at least one security policy. If you desire more detail or need more information about exactly what each parameter does, use the *show isakmp policy* command. This command expands on the previous command by spelling out each parameter and its current settings:

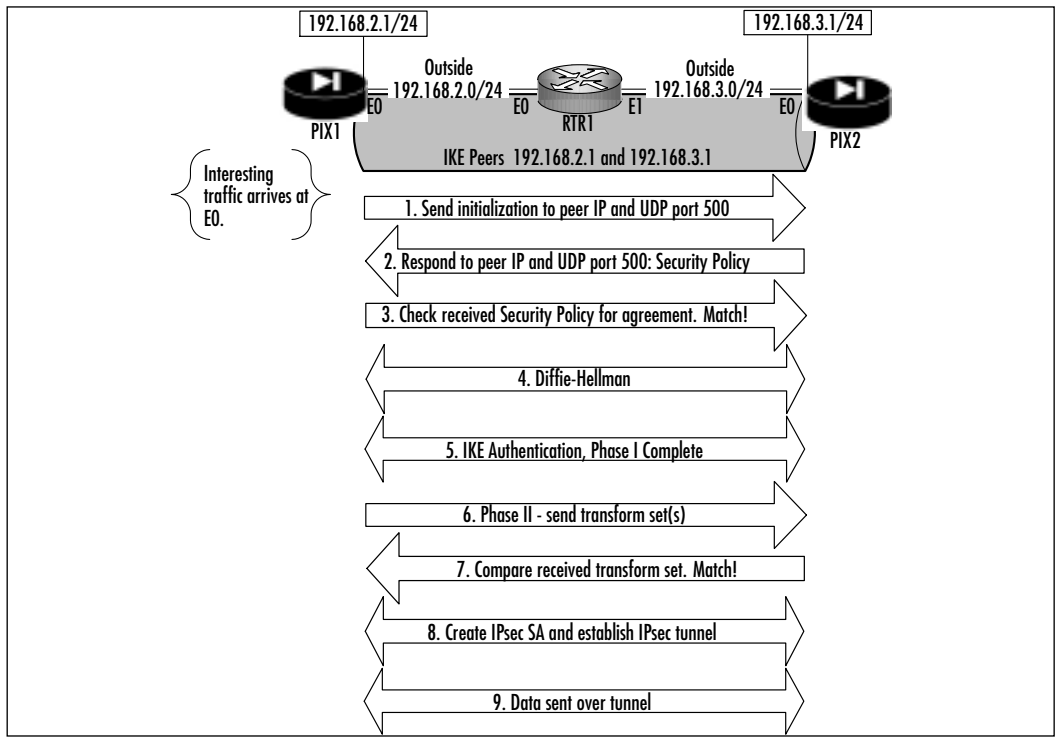
```
PIX1# show crypto isakmp policy
Protection suite of priority 99
     encryption algorithm:     DES - Data Encryption Standard (56 bit keys).
     hash algorithm:           Message Digest 5
     authentication method:    Pre-Shared Key
     Diffie-Hellman group:     #1 (768 bit)
     lifetime:                 9999 seconds, no volume limit
Default protection suite
     encryption algorithm:     DES - Data Encryption Standard (56 bit keys).
     hash algorithm:           Secure Hash Standard
     authentication method:    Rivest-Shamir-Adleman Signature
     Diffie-Hellman group:     #1 (768 bit)
     lifetime:                 86400 seconds, no volume limit
```

Another useful aspect of the *show crypto isakmp policy* command is that it shows you the default values that will be used if you do not specify any values. This information can be useful if you need to determine what a particular unspecified parameter would be if you do not configure it specifically.

IPsec cannot proceed unless IKE is working. The only exception is if you are not using IKE for IPsec—that is, you are using manually generated keys with IPsec.

If you want to watch the ISAKMP negotiation process between two IPsec peers, use the *debug crypto isakmp* command. This command generates a copious amount of output, so use it sparingly. You can use *debug crypto isakmp* to watch the IKE negotiation process and the exchange of session keys. The *debug crypto isakmp* command shows IKE going through Phases I and II. The entire process is triggered when interesting traffic (traffic that matches the applied crypto map) transits the IPsec protected interface. Once that happens, IKE contacts its peer, as shown in Figure 10.22. (Its source port and destination port will be UDP port 500, so you need to ensure that this port is allowed through.)

Figure 10.22 IKE Process



The first thing the peers do is validate that the hostname or IP address and key pair matches their configuration. The initiator sends its security policy parameters to the receiver, which then sends back parameters that match from its policy. Having agreed on the security policy, the IKE peers commence Phase I in earnest, completing the Diffie-Hellman and generating session keys. From there, IKE peer authentication is completed, finishing the Phase I security association. Phase II proceeds relatively quickly (hence the reason it is called "quick" mode) by negotiating the security policy that will be used to protect IPsec peer operations. Once Phase II is complete, IPsec then establishes the tunnel, and data transmission begins.

The most common problems that occur during the IKE phases are mismatched preshared keys and mismatched security policy parameters. The first step in troubleshooting IKE is to compare the configurations of each peer. You can do this with the commands we discussed previously. After you have ascertained that you have an IKE policy that will work on each firewall, initiate the IKE process after executing the appropriate *debug* command. That way, you can monitor its progress or lack thereof.

If you do not define an IKE security policy common to both peers or if you neglect to define a security policy at all, IKE will try the defaults for the various values. This means using DES for encryption, SHA for calculating the hash values, RSA for authentication, and Diffie-Hellman Group 1 (768 bits) with a lifetime of 86,400 seconds. Policy mismatches will be apparent when the output of the *show crypto isakmp sa* command shows "no state," meaning that the peers did not and could not negotiate main mode successfully due to the mismatch. The "no state" error also appears if there is key (password) disagreement between the two peers. Hash calculations will also fail, and this is something you can watch with the *debug crypto isakmp* command.

Cisco provides a *clear crypto isakmp sa* command that you can use to delete existing security associations and force a reinitialization. This command can be useful not only to clear an invalid security association, but it's also helpful in monitoring the IKE negotiation process with *debug*.

IPsec

After IKE successfully negotiates the parameters such as the method to be used for encryption, authentication, and the size key to use, IPsec is then ready to perform its mission of creating a VPN. IPsec requires that IKE already have negotiated the various previously identified parameters. IPsec peers compare transform sets to determine what each can support. They negotiate the authentication, encryption, and hash methods until they find agreement. If they do not find agreement, they do not become peers, and the tunnel will not be established.

To check which transform sets you have configured, use the *show crypto ipsec transform-set* command. Notice that this command tells you if IPsec will negotiate AH, ESP, or a combination of both. Here is an example:

```
PIX1# show crypto ipsec transform-set

Transform set FW1: { ah-md5-hmac  }
   will negotiate = { Tunnel,  },
   { esp-des esp-md5-hmac  }
   will negotiate = { Tunnel,  },
```

It is important for IPsec peers to have in their transform sets common parameters on which they can agree. Crypto maps are used to specify the traffic to be encrypted. Execute the *show crypto map* command to confirm your maps. For example:

```
PIX2# show crypto map

Crypto Map: "pixola" interfaces: {outside }

Crypto Map "pixola" 1 ipsec-isakmp
        Peer = 192.168.2.1
        access-list 100 permit ip 192.168.2.0 255.255.255.0 any (hitcnt=1)
        Current peer: 192.168.2.1
        Security association lifetime: 4608000 kilobytes/28800 seconds
        PFS (Y/N): N
        Transform sets={ pix, }
```

This command also identifies the IPsec peer and the interface to which the map is applied. In this example, PIX2 has the crypto map *"pixola"* applied to its outside interface. It is peering with PIX1 (at IP address 192.168.2.1) and will encrypt traffic that matches access list 100. It even tells you how many matches have been made against that access list—a quick way to determine if anything is being checked for IPsec processing.

After verifying that there is agreement in the transform sets and the crypto maps are defined correctly, confirm that data is actually being protected. To verify, use the *show crypto ipsec sa* command shown in Figure 10.23.

Figure 10.23 Verifying IPsec

```
PIX1# show crypto ipsec sa
interface: outside
Crypto map tag: pixola, local addr. 192.168.2.1

local ident (addr/mask/prot/port): (192.168.2.1/255.255.255.0/0/0)
remote ident (addr/mask/prot/port): (192.168.3.1/255.255.255.0/0/0)
current_peer: 192.168.3.1
PERMIT, flags={origin_is_acl,}

#pkts encaps: 5, #pkts encrypt: 5, #pkts digest 5
#pkts decaps: 5, #pkts decrypt: 5, #pkts verify 5
#pkts compressed: 0, #pkts decompressed: 0
#pkts not compressed: 0, #pkts compr. failed: 0, #pkts decompress
    failed: 0
```

Continued

Figure 10.23 Continued

```
#send errors 0, #recv errors 0

local crypto endpt.: 192.168.2.1, remote crypto endpt.: 192.168.3.1
path mtu 1500, ipsec overhead 56, media mtu 1500
current outbound spi: 3a18fca2
inbound esp sas:
spi: 0x61af4121(2451330208)

transform: esp-des esp-md5-hmac
in use settings ={Tunnel, }
slot: 0, conn id: 1, crypto map: pixola
sa timing: remaining key lifetime (k/sec): (4000159/9460)
IV size: 8 bytes
replay detection support: Y

inbound ah sas:
inbound pcp sas:

outbound ESP sas:
spi: 0x61af4121(2451330208)
transform: esp-des esp-md5-hmac
in use settings ={Tunnel, }
slot: 0, conn id: 1, crypto map: pixola
sa timing: remaining key lifetime (k/sec): (4000159/9460)
IV size: 8 bytes
replay detection support: Y

outbound ah sas:
outbound PCP sas:
```

The output of this command can be very abundant. The *crypto map* tag identifies the crypto map being used, whereas *local* and *remote "ident"* show the IP addresses of the local and remote peers. The *"pkts"* counters track how many packets have been encrypted, decrypted, and compressed. So far, five packets have been sent and received encrypted. This is an earmark of successful IPsec operation.

The crypto *"endpt"* section identifies the IPsec peers. Notice that the path MTU as well as the media MTU are shown, which can be useful in determining if fragmentation will occur. The SPI is a unique identification for this tunnel. We can also view the transform set parameters being used and whether it is operating in tunnel or transport mode. The *lifetime* indicates the amount of time left before the SA will be renegotiated. The last section, *"outbound sas,"* verifies that both inbound and outbound SA have been established. It also indicates how many seconds and kilobits are left before the SA must be renegotiated.

Check the SA lifetime with the *show crypto ipsec security-association* command. For example:

```
PIX1# show crypto ipsec security-association lifetime
Security association lifetime: 4608000 kilobytes/28800 seconds
```

You can use the *debug crypto ipsec* command to monitor IPsec negotiations, which will start once IKE is fully initialized between the peers. For ease of troubleshooting, run the two commands separately. Otherwise, you will be overwhelmed by the amount of data that they produce. First perform IKE troubleshooting (which has to occur before IPsec can proceed), and then move on to IPsec troubleshooting.

If you want to reinitialize IPsec, you can do so. This is useful when you want to clear corrupted or invalid sessions or if you want IPsec to establish a new tunnel. It can also be useful if you want to monitor IPsec operations from the onset using *debug* commands. At any time, you can manually force an SA negotiation to occur with the *clear crypto ipsec sa* command. The *clear crypto ipsec sa* command deletes existing security associations (all of them) and forces the establishment of new associations if there is an active trigger such as a crypto map. You can get very specific with this command, such as specifying a particular peer with *clear crypto ipsec sa 192.168.2.1.*

Capturing Traffic

Cisco has provided an excellent tool for capturing and analyzing network traffic with the introduction of PIX software version 6.2. When the *capture* command is used, the PIX can act as a packet sniffer on the target interface, capturing packets for later analysis. This command captures both inbound and outbound traffic.

Capturing packets that transit an interface is very useful for troubleshooting, because it enables you to determine exactly what traffic is being passed. When you're troubleshooting connectivity issues, it is often useful to capture packets from the incoming and outgoing interfaces. You can analyze the captured packets

to determine if there any problems with your configuration, such as IP address disagreement, or problems with IKE or IPsec, such as mismatched or expect parameters that are not being passed. Before this feature, the only recourse an engineer had was to install a packet capture device. The packet capture feature was introduced in PIX firewall version 6.2 and is only available for Ethernet interfaces. The syntax of the command is as follows.

```
capture <capture-name> [access-list <ID>] [buffer <bytes>] [ethernet-type
    <type>] [interface <if_name>] [packet-length <bytes>]
```

The first parameter, *capture-name*, defines a name for this particular capture session. All other parameters are optional. The *access-list* parameter specifies an access list to limit the source and destination of the traffic captured. By default, all IP packets are matched. The *buffer* parameter specifies the size of the buffer (in bytes) used to store captured packets. The maximum value is based on the amount of available memory on the PIX firewall. The default buffer size is 512K, and once the buffer fills up, the packet capture stops. The *ethernet-type* parameter specifies the protocols to capture. You can specify *ip, arp, rarp, ip6*, or any protocol number between 1 and 65535. By default, all Ethernet types are captured. (Setting the *ethernet-type* parameter to 0 specifies capturing all types.) The *interface* parameter specifies the interface on which to capture packets. The *packet-length* parameter specifies how much of each packet to capture. Usually for trouble-shooting, only the first few bytes of a packet are necessary, and the PIX captures up to 68 bytes. For example:

```
PIX1# capture inside-traffic access-list 100 buffer 20000 interface
    inside packet-length 200
```

In this example, we are capturing the first 200 bytes of traffic matching access list 100 on the inside interface. We have allocated 20,000 bytes for buffer storage of these captured packets.

Multiple traffic captures can be run simultaneously. To view the list of captures, use the *show capture* command. For example, the following command shows two simultaneous captures, *cap1* and *cap2,* being performed:

```
PIX1# show capture
capture cap1 interface inside
capture cap2 interface outside
```

To clear a capture buffer without stopping the capture, use the *clear capture* <*capture-name*> command. For example:

```
PIX1# clear capture cap1
```

To stop a capture and clear the associated buffer, use the *no capture <capture-name>* command. For example:

```
PIX1# no capture cap2
```

To stop a capture and save the associated buffer, use the *no capture <capture-name> interface <if_name>* command. For example:

```
PIX1# no capture cap1 interface inside
```

Displaying Captured Traffic

Cisco provides several options via which we can display our captured data. We can display it on the console, which provides for rudimentary viewing, or we can view it using a Web browser. We can even download our captured data and use third-party software such as Ethereal (www.ethereal.com) or tcpdump (www.tcpdump.org) to view them.

Display on the Console

In the course of troubleshooting a PIX firewall problem by capturing data, viewing the capture on the console is probably the most sensible option. If you opt to use the console for this purpose, it is best if you keep the packet-length short enough to get the primary headers (IP, TCP, etc.), because you can easily become confused scrolling through voluminous amounts of data on the simple textual console. To view a capture on the console, use the *show capture* command:

```
show capture <capture-name> [access-list <ID>] [count <number>] [detail]
    [dump]
```

If you have captured a great deal of data, you can filter it out by specifying an *access-list* in this command, which acts as a display filter. The *count* parameter is used to limit the number of packets displayed on the screen. The *detail* parameter increases the level of detail displayed. The *dump* parameter specifies that the data should be displayed in hex (this does not display MAC information). An example packet capture is displayed in Figure 10.24.

Figure 10.24 Packet Capture Example

```
PIX1# show capture inside-traffic count 6
71 packets captured
```

Continued

Figure 10.24 Continued

```
17:29:35.648434 192.168.2.1.23 > 192.168.2.2.11002: P 942178590:942178597
    (7) ack  2099017897 win 4096(fragment-packet)
17:29:35.848207 192.168.2.2.11002 > 192.168.2.1.23: . ack 942178597 win
    3531(fragment-packet)
17:29:37.610258 192.168.2.2.11002 > 192.168.2.1.23: P 2099017897:
    2099017898(1) ack 942178597 win 3531(fragment-packet)
17:29:37.610442 192.168.2.1.23 > 192.168.2.2.11002: . ack 2099017898 win
    4095(fragment-packet)
17:29:37.610686 192.168.2.1.23 > 192.168.2.2.11002: P 942178597:942178598
    (1) ack 2099017898 win 4096(fragment-packet)
17:29:37.808155 192.168.2.2.11002 > 192.168.2.1.23: . ack 942178598 win
    3530(fragment-packet)
```

Notice how the acknowledgments (ACKs) are incrementing. This particular capture was part of a Telnet session between 192.168.2.1 and 192.168.2.2; the 23 at the end of 192.168.2.1 tells you that it is the Telnet server. At this point, you should have a good idea just how useful *capture* can be in the troubleshooting process.

Display to a Web Browser

Cisco also makes it very easy to securely view packet captures (the packet headers in ASCII format) using a Web browser. To view the contents using your Web browser, enter the appropriate URL to the PIX firewall. The syntax is as follows:

```
https://pix_ip_address/capture/<capture-name>/
```

For example:

```
https://192.168.1.1/capture/inside-traffic/
```

Downloading Captured Traffic

The PIX firewall saves packet capture buffers in PCAP format, which can be downloaded and viewed with third-party software such as Ethereal or tcpdump. The capture can be downloaded either using HTTPS or TFTP. To download the file using HTTPS, enter the appropriate URL to the PIX firewall. The syntax is as follows:

```
https://pix_ip_address/capture/<capture-name>/pcap
```

For example:

```
https://192.168.1.1/capture/inside/pcap
```

This syntax downloads the packet capture to your client in PCAP format. Alternatively, you can download the file using TFTP. This is accomplished using the *copy* command on the PIX firewall. The syntax is as follows:

```
copy capture:<capture-name> tftp://<location>/<filename> [pcap]
```

Without the *pcap* keyword, the ASCII packet headers will be copied. With the *pcap* keyword, the binary file in PCAP format will be copied. For example:

```
PIX1# copy capture:inside-traffic tftp://192.168.99.99/pix-capture pcap
copying Capture to tftp://192.168.99.99/pix-capture:
```

In our example, we are copying the *inside-traffic* capture (in PCAP format) to the TFTP server at 192.168.99.99 to the *pix-capture* filename. Once the file has been copied, you can use any of the aforementioned software packages to open and analyze the captured packets.

Designing & Planning…

Support Options as Troubleshooting Tools

The PIX firewall can be a very critical device on your network. Network architecture planning needs to consider various support options to handle the loss or failure of your PIX firewall. Consider this troubleshooting by prevention, if you will. You can do it all yourself, farm out support to a third-party vendor (reseller), or purchase support from Cisco. Let's examine each option:

- In the "do it yourself" approach, you simply purchase the software and hardware, with no warranty or support other than what was provided as standard. If anything goes wrong, you need the knowledge and resources to fix it yourself.
- In the third-party option, you have a special arrangement with your vendor (reseller) to provide whatever you need to fix your problem, whether software or hardware. Although

Continued

your reseller might not have the depth and breadth of knowledge that Cisco does, as a reseller, it might be able to offer you a substantial discount on support.

- Using Cisco via the SMARTnet program can ensure that you always have access to a large pool of expert knowledge and the "latest and greatest" information regarding configuration, troubleshooting, and bug fixes. The Cisco Web site offers a wealth of tools and information that you can use to aid your troubleshooting. You can also opt to acquire the Cisco Connection Online (CCO) membership to gain access to even more support such as the ability to open or browse TAC cases online. SMARTnet also provides hardware replacement and software upgrades.

Two things can break on your PIX firewall: the software or the hardware. To protect against hardware failures, you have the option of stockpiling spares. Depending on the ratio of active to stock units, this choice could be cost prohibitive. Software can be plagued with bugs that you discover after you have deployed the perfect configuration. Certain commands or features might not work as you want them to or not work at all. In any case, you will require information from Cisco to work around the problem or access to the latest release of software that fixes your problem. In general, you are better off putting your firewall under a SMARTnet maintenance contract with Cisco to ensure that you always have access to the latest releases of software. Software is generally much more difficult to fix on your own than hardware, which you can easily replace in case of a failure. You definitely cannot rewrite the software code to fix a problem, and you'll end up spending an excessive amount of time developing a workaround to a problem caused by a buggy software release.

Monitoring and Troubleshooting Performance

We mentioned previously the importance of matching the model of PIX firewall you deploy to the demands you place on it. You need to consider several factors in addition to the amount of traffic you are passing. Table 10.3 summarizes the loads that each model can handle, including information about encryption. Ensure that your design considers these load limits.

Table 10.3 PIX Firewall Model Features and Capabilities

Model	Hardware Maximums (CPU/SDRAM /FLASH)	Cleartext Through-put	DES IPsec Through-put	3DES IPsec Through-put	Simul-taneous VPN Tunnels
501	133MHz AMD SC520 16MB RAM 8MB Flash	10Mbps	6Mbps	3Mbps	5 peers
506 (EOS)	200MHz Intel Pentium 32MB RAM 8MB Flash	20Mbps	20Mbps	10Mbps	25 peers
506E	300MHz Intel Celeron 32MB RAM 8MB	20Mbps	20Mbps	16Mbps	25 peers
515 (EOS)	200MHz Intel Pentium 32MB RAM 8MB Flash	146Mbps	20Mbps	10Mbps	25 peers
515E	433MHz Intel Celeron 64MB RAM 16MB Flash	188Mbps	33–120Mbps	63Mbps UR 22Mbps R	2,000
520 (EOS)	350MHz Intel Celeron 64MB RAM 16MB Flash	370Mbps	20Mbps	10Mbps	*
525	600MHz Intel Pentium III 256MB RAM 16MB Flash	360Mbps	120–140Mbps	70Mbps	2,000
535	1GHz Intel Pentium III 1GB PC133 RAM 16MB Flash	1Gbps	200Mbps	100Mbps	2,000

Cleartext throughput means unencrypted data passing through the firewall, while IPsec (DES and 3DES) throughput is considered encrypted. The cleartext throughput of the PIX firewall ranges from a low of 10Mbps to a high of 1Gbps.

Three key components of the PIX firewall that affect performance are the CPU, memory, and network interfaces. You need to understand how to monitor these components and ensure that their load is not reaching the limits. We discuss the monitoring of these three components in the following sections. The ultimate question is, can your firewall handle the loads you will place on it?

CPU Performance Monitoring

Your CPU does it all: passes traffic, creates VPN tunnels, and performs encryption on demand. The rule of thumb is that during normal operational mode, the CPU load should stay below 30 percent, on average. During peak traffic hours and attacks, you will see the CPU surge up higher, but that is normal. However, if the CPU utilization consistently stays above 30 percent with normal network activity, consider upgrading to a more powerful model.

Many functions can tax CPU, but encryption (DES and 3DES) has the biggest potential to consume your CPU's precious time. If you are going to deploy a large number of encrypted tunnels (VPNs), we recommend you monitor the processor carefully. If utilization goes high, consider adding a card to the PIX to handle VPN functions (the VPN Accelerator Card). Alternatively, you can think about offloading VPN functions from the PIX to a dedicated VPN concentrator (such as the VPN 3000 series from Cisco). The amount of traffic passing through the firewall is also a factor. If you are seeing high traffic utilization, monitor the CPU utilization on a regular basis to ensure that it is not peaking. The best way to do this is to use a tool such as MRTG or HP OpenView to monitor the CPU through SNMP. See Chapter 6 for details on how to do this.

Logging and the excessive use of *debug* commands also affect CPU utilization. To avoid consuming precious CPU cycles, you should set logging to the minimum level of information that you actually need. Table 10.4 displays the logging levels you have at your disposal. If there is a reason you need high logging levels, consider turning off log messages that you do not need using the *no logging message*. See Chapter 6 for detailed information on logging.

Table 10.4 Logging Levels

Description	Numerical Value
Emergency	0
Alert	1
Critical	2
Error	3
Warning	4
Notification	5
Informational	6
Debugging	7

You can determine the logging options and levels that are enabled on a PIX firewall using the *show logging* command. For example, on this firewall, all logging is disabled:

```
PIX1# show logging
Syslog logging: disabled
    Facility: 20
    Timestamp logging: disabled
    Standby logging: disabled
    Console logging: disabled
    Monitor logging: disabled
    Buffer logging: disabled
    Trap logging: disabled
    History logging: disabled
```

The show cpu usage Command

The *show cpu usage* command provides a snapshot of the short-term CPU utilization statistics. Although this information is not useful for history or trending purposes, it can immediately inform you if the CPU is overloaded at the time the command is executed. This command does allow you to check in real time if the CPU is the cause of any performance degradations. For example:

```
PIX1# show cpu usage
CPU utilization for 5 seconds = 2%; 1 minute: 1%; 5 minutes: 1%
```

If you suspect that IPsec encryption is causing performance degradation, use this command before turning on encryption to take a baseline of CPU

utilization. Then enable IPsec and run the command again. Compare the CPU utilization. Run the command a few times over a time interval to ensure that the data you gathered is accurate.

The show processes Command

If CPU utilization is high, you will often need much more detail than the *show cpu usage* command provides. This is where the *show processes* command comes in. This command identifies every process running on the PIX firewall, how much memory is it using, and CPU cycles. This information is collected from the time the PIX firewall was started. As shown in Figure 10.25, the output of this command is voluminous; in fact, this is an abbreviated listing of the actual output from a PIX 501 firewall, the lowest end of the PIX firewalls. Here we do not explain every row of this of the display, but we do discuss how to interpret what you see in the columns. A detailed analysis of this command is available on Cisco's Web site at www.cisco.com/warp/public/110/pix_shproc.html.

Figure 10.25 Output of the show processes Command

```
PIX]# show processes

     PC        SP        STATE      Runtime     SBASE     Stack Process
Hsi 800b0e09 80759798 8052ddd8           0 80758810 3532/4096 arp_timer
Lsi 800b5271 8077c880 8052ddd8           0 8077b908 3912/4096 FragDBGC
Lwe 8020685d 808b8e20 80507300           0 808b6ed8 7644/8192 Logger
Hwe 8020a550 808bbee8 805075b0           0 808b9f70 8008/8192 tcp_fast
Lsi 80137edd 809400f0 8052ddd8           0 8093f168 3928/4096 xlate clean
Lsi 80256f4d 8096c430 8052ddd8           0 8096b4a8 3900/4096 route_process
Mwe 800d2671 809b19e0 8052ddd8           0 809afa68 6940/8192 IPsec timer
Lwe 8012ff5a 809daae8 80539908           0 809d9e50 3704/4096 pix/trace
Lwe 8013016a 809dbb58 80539fd0           0 809dace0 3704/4096 pix/tconsole
Hwe 800b2dd0 809ddbe8 80753b9c           0 809dbd70 7196/8192 pix/intfl
H*  80015207 7ffffe2c 8052ddc0         200 809e1ea0 12652/16384 ci/console
Csi 801299b3 809e6e88 8052ddd8          10 809e5f30 3440/4096 update_cpu_usag

A    B        C        D          E          F          G          H
```

The first character in Column A refers to the priority of the process, which can be ranked (highest priority to lowest priority): **C**ritical, **H**igh, **M**edium, or **L**ow. The next two characters in Column A refer to the current operating state of the process, which can be any of the values shown in Table 10.5.

Table 10.5 Process Operating States

Value	State	Description
*		Currently running.
E	Waiting	For an event to occur or complete (i.e., access-list or monitoring)
S	Ready to run	Gave up processor time slice (idle)
rd	Ready to run	Conditions for activation have occurred
we	Waiting	Watching for certain activity to occur
sa	Sleeping	Until a set time occurs, like 2 PM
si	Sleeping	For a set time
sp	Sleeping	For a set time (alternate)
st	Sleeping	Time expiration will activate
hg	Hung	Crash or error; eventually will clear
xx	Dead	Terminated and to be deleted

Columns B and C are the counter and stack pointers for the process in question. This is basically how the process is differentiated from other processes. Column D identifies the thread queue used by the process, and this queue is shared with other processes. The runtime value in Column E is how much CPU time in milliseconds the process has consumed since it started. The SBASE column denoted by F shows the starting address space for the process, and Column G shows the ratio of used/total available space in bytes allocated to the process. Bad processes will attempt to invade the space used by other processes. Column H identifies the process name.

This command is very useful to determine the processes that are taking up too many CPU cycles. To figure this out, issue the *show processes* command twice, waiting about 1 minute in between. For the process that you suspect to be a problem, subtract the Runtime value from the second command from the Runtime value from the first command. The result indicates the amount of CPU time (in milliseconds) the processed has received during that 1 minute. It is important to understand that some processes are scheduled to run at certain times, and others only run when they have information to process. The 577poll process typically has the largest runtime of all processes because this is the one that polls Ethernet interfaces to see if they have any data that needs to be processed.

The show perfmon Command

One extremely useful command for performance monitoring on the PIX firewall is the *show perfmon* command. It shows details a number of statistics, including translations, connections, fixup, and AAA. This is the only command that you can use to view the "average" values for the number of translations and connections on the firewall. The nice thing about this command is that it breaks the connections down by protocol, as shown in the output in Figure 10.26. This breakdown can help you determine if a particular connection is using up too much CPU or memory. Table 10.6 lists the values in the *show perfmon* command.

Figure 10.26 Output of the show perfmon Command

```
PIX1# show perfmon

PERFMON STATS:     Current       Average
Xlates                0/s            0/s
Connections           0/s            0/s
TCP Conns             0/s            0/s
UDP Conns             0/s            0/s
URL Access            0/s            0/s
URL Server Req        0/s            0/s
TCP Fixup             0/s            0/s
TCPIntercept          0/s            0/s
HTTP Fixup            0/s            0/s
FTP Fixup             0/s            0/s
AAA Authen            0/s            0/s
AAA Author            0/s            0/s
AAA Account           0/s            0/s
```

Table 10.6 Values in the show perfmon Command

Parameter	Description
Xlates	Translations built up per second.
Connections	Connections established per second.
TCP Conns	TCP connections per second.
UDP Conns	UDP connections per second.

Continued

Table 10.6 Continued

Parameter	Description
URL Access	URLs (Web sites) accessed per second.
URL Server Req	Requests sent to Websense/N2H2 per second (requires the *filter* command).
TCP Fixup	Number of TCP packets that the PIX forwarded per second.
TCP Intercept	Number of SYN packets per second that have exceeded the configured embryonic limit.
HTTP Fixup	Number of packets destined to port 80 per second (requires the *fixup protocol http* command).
FTP Fixup	FTP commands inspected per second.
AAA Authen	Authentication requests per second.
AAA Author	Authorization requests per second.
AAA Account	Accounting requests per second.

As with any measurement, if you do not have a baseline, this type of information is useless. Execute the command on a regular basis over time to build a baseline. You can then compare values to this baseline to find anomalies.

Memory Performance Monitoring

Memory utilization can be as important in determining performance as the CPU. Flash memory is used to hold the PIX operating system and configuration. Unless you have a very large software image, allocation issues with flash memory are not a concern. Main memory, the focus of this section, is the working space of the PIX firewall. When the PIX firewall first boots, it initializes and runs the OS from flash memory. It does not load the OS into main memory, which is used for data only. The memory is also used for all processes as well as buffering incoming and outgoing traffic. Because it is used by so many different aspects of the firewall, it is critical to ensure that you have enough memory. You can run several commands to help with this task. In addition, similar to CPU utilization, we recommend using an SNMP tool such as MRTG or HP OpenView to monitor the amount of available memory on the PIX firewall.

The show memory Command

The *show memory* command provides an easily comprehensible overview of how much memory is installed and how much is currently being used. This command

simply shows you the amount of total and free memory at the time that you run the command. Here is an example of the *show memory* command:

```
PIX1# show memory
16777216 bytes total, 4517888 bytes free
```

To optimize the usefulness of this command, run it on a PIX firewall that has a very basic configuration. That is, run it on a PIX firewall that is not running encryption or other functions and record that information. Then as you add features, execute the command and compare the output. Doing so enables you to record approximately how much memory is consumed by each process.

The show xlate Command

One process that consumes memory is address translation. Each translation requires approximately 56 bytes of memory. Knowing this, you can run the *show xlate* command. For example:

```
PIX1# show xlate
100 in use, 341 most used
```

Multiply the number of translations by 56 bytes to determine how much memory has been consumed for translations. In our example, we have 100 translations in use, which means we have 5600 bytes of memory allocated for translation alone.

The show conn Command

Each connection made to the firewall also consumes memory. The amount of memory consumed depends on the type of connection. A UDP connection consumes 120 bytes; a TCP connection requires 200 bytes. This memory consumption is necessary to build the connection and maintain state information. Here is an example of the *show conn* command:

```
PIX1# sh conn
2 in use, 2 most used
```

If we have 100 TCP connections made through this PIX firewall, that will require 20K of main memory. Of course, this is a transitory number and will fluctuate depending on the times of the day.

The show block Command

The PIX firewall reserves certain amounts of memory to handle special traffic after the configuration is loaded and running and before any other memory

allocation occurs. Certain amounts of memory are allocated into variable byte-sized blocks. Predefining such set-sized blocks relieves the firewall from having to carve memory on the fly. You can use the *show blocks* command to view the currently set block sizes. For example:

```
PIX1# show blocks
  SIZE     MAX     LOW     CNT
     4    1600    1563    1600
    80     400     386     400
   256     500     143     500
  1550    1700    1102    1315
 16384       8       8       8
```

We need to clarify the output of this command starting with the SIZE column, which is measured in bytes. The 4-byte blocks are reserved for certain traffic types such as DNS, IKE, TFTP (traffic that is small and bursty). The 80-byte blocks are used to store failover hellos and TCP intercept acknowledgements. The 256-byte blocks store stateful failover messages. The 1550-byte blocks support Ethernet (10 and 100) packets as they pass through the firewall. The 16384-byte blocks will never be used unless you have Gigabit Ethernet interfaces, something you will only see on the high-end firewalls.

The MAX column identifies the maximum number of each type of memory blocks available. The LOW column indicates the lowest number of blocks that have been available since the firewall booted. Stated mathematically, subtract LOW from MAX to get the maximum number of blocks that were used at any particular time. The CNT column shows the available number of blocks. Use the *clear blocks* command to reset the LOW and CNT counters.

Network Performance Monitoring

Congested network interfaces can degrade overall performance. You need to ensure that the interfaces on your PIX firewall can handle the demands placed on them. Cisco offers several commands to check the status of your interfaces.

The show interface Command

One such command is **show interface**. You can check how much bandwidth is being consumed and check a myriad of error counters. We have discussed **show interface** previously in the chapter, and will not rehash what has already been covered.

The show traffic Command

You can narrow your focus to capture the specific number of packets and bytes that are transiting each interface on the PIX firewall. The *show interface* command provides similar information, but you have to make it a specific point to zoom in on that information to determine exactly the amount of traffic being passed on a per-interface basis.

The *show traffic* command provides statistics on the number of packets and bytes passed through each interface. As you can see in the output in Figure 10.27, *show traffic* tells you how long the interface has been in operation (either the firewall below has been in operation almost three hours or that much time has elapsed since the clearing of the statistics). The command output displays the amount of traffic transmitted and received in that amount of time.

Figure 10.27 Output of the show traffic Command

```
PIX1# show traffic
outside:
        received (in 10035.150 secs):
                2 packets        678 bytes
                0 pkts/sec       0 bytes/sec
        transmitted (in 10035.150 secs):
                14 packets       1026 bytes
                0 pkts/sec       0 bytes/sec
inside:
        received (in 10035.150 secs):
                0 packets        0 bytes
                0 pkts/sec       0 bytes/sec
        transmitted (in 10035.150 secs):
                15 packets       900 bytes
                0 pkts/sec       0 bytes/sec
```

You can reset the traffic counters using the *clear traffic* command, which resets the counters to 0.

Identification (IDENT) Protocol and PIX Performance

There is one particular protocol that we need to address because it affects PIX performance. This is the identification protocol specified in RFC 1413. The purpose of this protocol is to enable HTTP, FTP, or POP servers to confirm the identity of clients. When a client connects to one of these ports, a server running IDENT will attempt to connect to TCP port 113 on the client. If successful, the server will read certain identifying data from the client machine. In theory, this process would reduce spam or illegitimate usage by forcing users to connect from legitimate sources. In practice, the IDENT protocol can be circumvented easily.

Users behind a PIX firewall are protected from IDENT by default. Since the IDENT protocol provides information about the user, it can provide details about the internal network, which can be a violation of your security policy. The PIX firewall, like any good firewall, prevents this passage of internal details to the outside. However, the downside of this protection is that users could perceive a very noticeable delay in the server responding to their requests as it attempts to check their identities, or they could even experience a total lack of response.

To identify IDENT issues, set logging to the debugging level. Once that is turned on, you will see denied TCP attempts to port 113 attempts. To get around this issue, you have the following choices:

1. You can contact the administrator of the server running IDENTD and have it turned off. However, you will have to do this for *each* server that has this problem.
2. You can pass IDENT traffic through your firewall unmolested by permitting it with access lists or conduits. This would pass internal network details to the outside, which can compromise security.
3. Another (the recommended) solution is to use the *service resetinbound* command. This command sends a TCP reset (RST) to the IDENT server, which essentially tells it that the client does not support IDENT. Upon receiving that reset, the server provides the requested service to the user. Once this command is entered, the PIX firewall starts sending resets to traffic not permitted by the security policy rather than dropping it silently and causing the user to incur a time penalty.

Summary

This chapter introduced a troubleshooting methodology based on the OSI model. Using this approach, you start at the lowest layers and work up the stack. Doing this enables you to eliminate lower (and typically simpler) layer causes before focusing efforts on higher (and typically more complex) layer aspects of PIX firewall troubleshooting.

Knowledge is power! Knowing the various models of PIX firewalls and their capabilities is extremely important to troubleshooting. Certain models of the PIX firewall, such as model 501 and 506, do not support failover. Knowing such details would prevent you from wasting your time attempting to solve problems with features not supported on a particular model. Other useful information to know about the PIX firewall includes the number of supported connections as well as the number and types of NICs supported (such as Token Ring and Ethernet).

Although the PIX firewall supports a limited number of network types, familiarity with the cables used to connect to those networks can be a useful asset to troubleshooting. The PIX firewall uses standard TA586A/B wiring schemes for 10/100 Ethernet, and SC multimode fiber optic cables for Gigabit Ethernet. The failover cable is an instance of a specialized function made possible by adhering to a stringent Cisco proprietary wiring scheme.

In order for the PIX firewall to perform its function, it must be able to service its internal networks as well as know how to forward traffic to the appropriate destination. This is made possible using a static routes or RIP. You need to be able to troubleshoot and resolve reachability issues to enable the PIX firewall to perform its job.

Translation is required for providing connectivity through the PIX firewall. Your troubleshooting toolbox includes many Cisco commands such as *show xlate, show nat,* and *show global*, all used to check translation configurations and operations. Ensure that you make *clear xlate* a regularly executed step in your troubleshooting, especially after making configuration changes.

Other connectivity issues you need to troubleshoot involve ensuring that only the proper access is granted to certain external networks. You can use commands such as *show conduit, show access-list,* and *show access-group* to validate what access is granted.

IPsec is probably one of the most complex features you will ever configure on the PIX firewall. The troubleshooting is equally complex. In this chapter, we covered several of the most critical commands available for validating IPsec

operation. When troubleshooting, divide your efforts to enable better focus by first troubleshooting and resolving IKE issues, then focusing on IPsec. IPsec depends on IKE, but IKE does not need IPsec to perform its functions.

With the introduction of PIX version 6.2, Cisco has provided a useful packet capture and analysis tool in the form of the *capture* command. This command allows you to troubleshoot networks remotely by enabling the capture and analysis of networks connected to the PIX firewall. This reduces the need to install a third-party device on the target network to obtain information about it.

The best troubleshooting practice is proactive monitoring to detect problems before they become unmanageable. You can accomplish this proactive state by gathering performance data about various aspects of your PIX firewall such as CPU performance, memory consumption, and network bandwidth utilization statistics.

Solutions Fast Track

Troubleshooting Hardware and Cabling

- ☑ Use the OSI model to guide your troubleshooting efforts, starting with Layer 1 (the physical layer) and working your way up.
- ☑ There are several models of the PIX firewall, starting with the fixed configuration 501 up to the current top-of-the-line model, the configurable 535. In addition, a Firewall Services Module (FWSM) is available for the Catalyst 6500 series switches.
- ☑ Monitoring the PIX firewall POST can provide valuable information about the hardware that is installed.
- ☑ The *show interface* command provides very useful statistics about network interfaces, which can provide clues about network malfunctions.
- ☑ The PIX firewall uses the TA586A/B wiring standard for connectivity to 10/100 Ethernet networks. Gigabit Ethernet networks connect to the PIX firewall via SC multimode fiber optic cables.

Troubleshooting Connectivity

- ☑ Like any network device, the PIX firewall must know how to reach its destinations.
- ☑ The PIX firewall can use static routes or RIP.
- ☑ The routing troubleshooting process is similar to the steps that you would perform on a router (checking routing tables, verifying next-hop reachability, and so on).
- ☑ IP address assignment should be one of the first items that is checked for correctness before any other item.
- ☑ Since translation configuration has as much effect on connectivity as routing, you need to know how to validate it.

Troubleshooting IPsec

- ☑ IPsec troubleshooting needs to be very methodical due to its complexity.
- ☑ Validate IKE first before attempting to tackle IPsec itself, since IPsec depends on IKE for its operation.
- ☑ The *show isakmp* command can give a quick snapshot of IKE configuration on the firewall, enabling you to check the parameters for correctness.
- ☑ The *show crypto ipsec* command with select keywords can enable you to check various aspects of IPsec, such as its security associations and transform sets.

Capturing Traffic

- ☑ Cisco introduced the *capture* command in PIX version 6.2.
- ☑ This command enables you to remotely capture packets of networks connected to the PIX firewall.
- ☑ Captured packets can be viewed on the console, viewed or downloaded from a Web browser, or downloaded to a workstation via TFTP for analysis by third-party software such as tcpdump.

Monitoring and Troubleshooting Performance

- ☑ Proactive monitoring can prevent problems from becoming unmanageable.
- ☑ CPU performance and memory consumption can be indicators of problems.
- ☑ The *show processes* command can help identify the processes that are running and the ones that could be consuming more PIX firewall resources than they should.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: Which PIX firewall models support Gigabit Ethernet?

A: 525 and 535.

Q: I suspect a key mismatch between my IKE peers. What can I do to verify that?

A: You can check syslog messages, which will display information about these types of errors. You can also use *show crypto isakmp* and view the configuration.

Q: What is the latest version of PIX software that supports Token Ring and FDDI interfaces?

A: Version 5.3. versions after that have no support for Token Ring or FDDI.

Q: How do I determine how much memory is installed on my PIX firewall?

A: Use either the *show version* command or the *show memory* command.

Q: In a failover configuration, what determines which firewall is active and which is standby?

A: The failover cable that Cisco provides is strange, such that one end will cause the firewall to become the active in a failover configuration whereas the other end will become standby.

Q: Where is the configuration file for the PIX firewall stored?

A: It is stored in flash memory. No NVRAM like that is commonly found on Cisco routers.

Q: What routing protocols does the PIX firewall support?

A: At the time of this writing, only RIP versions 1 and 2 are supported.

Index

A

AAA, 218–228
  See also authentication; authorization; accounting
aaa accounting command, 272
aaa authentication command, 261
aaa authentication enable console command, 249
aaa authorization command, 251, 253, 260, 271
AAA Floodguard, 191–194
AAA protocols, 223–228
aaa-server command, 244, 261, 393
AAA server. *See* Cisco Secure Access Control Server for Windows
AC (address concentrator), 209
access attacks, 8
access control, 4, 13, 207–209
access control lists (ACLs), 137, 166
access-group command, outbound traffic and, 101, 105
access-list commands, 117, 276, 362
  outbound traffic and, 101, 106
access-list statement, 117
access lists
  case study and, 124–126
  configuring, 100
  crypto, 362–364
  downloadable, 275–281
  inbound traffic and, 113
  outbound traffic and, 100–109
  troubleshooting, 583–588
access modes, 72–75
access rules, 512–519
accounting, 218, 223
  configuring, 272–275
ACLs (access control lists), 137, 166
activation key, 65–67
  DES, for PDM configuration, 456
activation-key command, 66
active code filtering, 173–175
Active Directory, ILS protocol for, 164
active hosts, 185
active mode, 141–146
ActiveX applets, filtering, 173–175
  web filtering and, 166
Adaptive Security Algorithm (ASA), 46–55, 92
  how it works, 49
  packets processed by, 138
address concentrator (AC), 209
address translation, 26–29, 57, 93
  troubleshooting, 580–583
administrative access modes, 72–75
AH protocol, 335–337
alarm action, 179
alias commands, 146, 147
American Registry for Internet Numbers (ARIN), 97
Apple QuickTime, 153, 155
application inspection, 138–165
application proxies, 22
apply command, 109–111
  case study and, 127–129
ARIN (American Registry for Internet Numbers), 97
ASA. *See* Adaptive Security Algorithm
attacks, 6, 7
  classifications for, 178
attribute-value (AV) pairs, 222
auditing, 4, 13
  configuring, 179–182
auth-prompt command, 269
authentication, 4, 13, 218
  configuring
    console authentication, 242–250
    through firewall, 260–270
  for NTP, 325
  for SSH, 309
authentication algorithms, IPsec and, 347
authentication header (AH) protocol, 335–337
authentiction, 221
authorization, 4, 218, 222
  configuring, 270–272
    on firewall, 250–260
authorization timers, 262
AV pairs, 222
availability, 4, 13, 59

B

backups, ensuring system availability and, 4
biometrics, 4
buffered logging, 292

C

ca commands, 307, 356–362
CA. *See* certificate authority
cabling, troubleshooting for, 555–570
Cain & Abel program, 69
CAN-2002-0954, 69
capture command, 597
capturing traffic, 597–602
  downloading and, 600
case study on configuring PIX firewalls, 122–129
CCIE (Cisco Certified Internetwork Expert Security), 32–34
certificate authority (CA), 321

support for, 348, 354–362
certificate revocation list (CRL), 321
certifications, 31–36
Challenge-Handshake Authentication Protocol (CHAP), 209, 210
checksums, 3
cipher order, setting preferred, 308
Cisco
PIX Device Manager. *See* PIX Device Manager
PIX Management Information Bases (MIBs), 317
PIX. *See* PIX firewalls
Cisco Certified Internetwork Expert Security (CCIE), 32–34
cisco default password, 70
Cisco IP Phones
phone placed hold and, 164
special DHCP options for, 189
SSCP protocol for, 161
Cisco IP/TV, 153, 155
Cisco PDM. *See* PIX Device Manager
Cisco Secure Access Control Server for Windows (AAA server), 218, 228–242
installing/configuring, 230–237
products for supported by PIX firewall, 228
Cisco Secure IDS appliance, 175, 178
Cisco Secure PIX Firewall Advanced exam (CSPFA), 34–36
Cisco Secure Policy Manager (CSPM), 450
Cisco Security Specialist 1(CSS-1), 31
Cisco Security Wheel, 11–17
Cisco software VPN client, support for, 390–406
Cisco TFTP servers, 68
Cisco type 5 hashes, 69
Cisco type 7 passwords, 69
CiscoWorks, 45, 316
cleartext passwords, RIP protocol and, 201
CLI. *See* command-line interface
clock timezone command, 323
clock. *See* date and time
closed systems, 2
command authorization, 250–260
See also authorization
command-line interface (CLI), 71–82, 304
basic commands for, 75
community strings, 316, 318
conduit command, 113
conduits
case study and, 127–129
inbound traffic and, 113
troubleshooting, 583–588
confidentiality, 3, 13
configuration mode commands, 317
configuration replication, 417
configurations
case study illustrating, 122–129
factory default, 71
managing, 79–82
configure command, 81
configure terminal command, 74
conn tables (connection tables), 48, 137
connected routes, 197–199
connection tables (conn tables), 48, 137
connectivity, troubleshooting, 570–588
console authentication, 242–270
See also authentication
console logging, 293
console port, 63–65, 304
content filtering, 166
control connection, FTP protocol and, 141
Coordinated Universal Time (UTC), 321
copy command, 80
CPU, performance monitoring for, 604–609
CRL (certificate revocation list), 321
crypographic tokens, 4
crypto access lists, configuring, 362–364
crypto dynamic-map commands, 385
crypto map command, 345, 362, 366
crypto maps
configuring, 366–369
dynamic, 384–386
cryptography algorithms, IPsec and, 346
CSPFA (Cisco Secure PIX Firewall Advanced exam), 34–36
CSPM (Cisco Secure Policy Manager), 450
CSS-1 (Cisco Security Specialist), 31
CU-SeeMe/CU-SeeMe Pro, 161
cut-through proxy, configuring, 260–265
CVE vulnerability CAN-2002-0954, 69

D

datastreams, 46
date and time
checking, 322
configuring, 321–326
time zone, displaying, 323
daylight savings time, 322
db-dmz interface, 123
DDoS attacks (distributed DoS attacks), 8
debug commands
for DHCP clients, 185
for multicasting, 206
debug ssh command, 311
default route, 78
demilitarized zones (DMZs), 24–26
denial of service attacks (DoS attacks), 8
deny statement, IPsec and, 345
DES activation key, for PDM configuration, 456

DHCP (Dynamic Host Configuration Protocol), 182–189
dhcp auto_config command, PPPoE and, 211
DHCP clients, 183–185
  debug commands for, 185
DHCP servers, 185–189
dhcpd commands, 186–189
digital certificates, 321
distributed DoS (DDoS), 8
dmz interface, 123
DMZs (demilitarized zones), 24–26
DNS Guard, 146–148
DNS. *See* Domain Name Service
domain-name command, 76, 356
Domain Name Service (DNS), application inspection for, 146–148
domain names, for DHCP clients, 186
DoS attacks (denial of service attacks) , 8
drop action, 179
dynamic address translation (dynamic NAT), 28, 93–100
Dynamic Host Configuration Protocol (DHCP), 182–189

E

egress filtering, 194
emacs-style commands, 75
embryonic connections, 192
enable mode, 73
  password for accessing, 78
enable password, 78
encapsulating security payload protocol, 335, 337
Enterprise Firewall, 23
Enterprise Security Manager (ESM), 16
ESP protocol, 335, 337
established udp command, 164
ethical hackers, 16
External Data Representation (XDR) format, 152
external threats, 6

F

facilities for logging, 302
factory default configurations, 71
failover, 296, 413–447
  cable for, troubleshooting, 578
  LAN-based, 434–443
    disabling, 443
  manual, for maintenance, 443
  standard, 422–433
    disabling, 433
  stateful, 420
failover commands, 419, 421, 425–443
failover licensing, 67
fault tolerance, 59
File Transfer Protocol (FTP)
  application inspection for, 141–146
  logging for, 301
  proxy for, 56
filter activex command, 175
filter command, 150, 168
  long URLs, options for, 170
filter java command, 174
filtering policy, configuring, 168
filtering servers, 167–169
  caching replies from, 170
filtering web traffic, 165–175
  configuring policy for, 168
  fine-tuning the process, 169–172
firewall interfaces, 23–26
Firewall Toolkit (fwtk), 18
firewalls, 17–31
  proxy-based, 22, 23
  types of, 19–23
  See also PIX firewalls
fixup command, 55
fixup protocol command, 140
fixup protocol ftp command, 143
fixup protocol h323 command, 160
fixup protocol http command, 150
fixup protocol ils command, 165
fixup protocol rtsp command, 155
fixup protocol sip command, 161
fixup protocol skinny command, 161
fixup protocol smtp command, 149
fixup protocol sqlnet command, 158
flash activation key, 66
floodguard command, 192
flooding, mechanisms to combat, 191–194
FragGuard feature, 189–191
fragment commands, 190
fragmented packets, 189–191
FTP logging, 301
FTP proxy, 56
FTP. *See* File Tranfer Protocol
fwtk (FirewallToolkit), 18

G

gatekeepers, H.323 protocol set and, 161
Gauntlet (Secure Computing), 23
global addresses, 27
global command, 93
GMT (Greenwich Mean Time), 321
graphical user interfaces (GUIs), PIX firewall support for, 450
graphing applications, 317
graphs, 537–547
Greenwich Mean Time (GMT), 321
GUIs (graphical user interfaces), PIX firewall support for, 450

H

H.323 protocol set, 159–161
hardware, troubleshooting, 555–570
hashes, 3
hashing algorithms, 347
high availability, 59, 414
host groups, 202
hostname command, 76, 356
hot spare, 84
HP OpenView, 316
HTML, filtering and, 173–175
HTTP. *See* Hypertext Transfer Protocol
Hyperterm, 63
Hypertext Transfer Protocol (HTTP)
  application inspection for, 150
  virtual http command and, 266–268

I

icmp command, 114
ICMP traffic, 114
icmp-type object groups, 118
IDENT protocol (identification protocol), performance and, 613
identity NAT, 97–100
IDS signatures (list), 176–178
IGMP (Internet Group Management Protocol), 203, 205
IKE policies, 352–354
  pre-shared key for, 354
IKE. *See* Internet Key Exchange
ILS (Internet Locator Service), application inspection for, 164
inbound traffic, 111–116
information security/insecurity, 3–8
ingress/egress filtering, 194
initial sequence number (ISN), 54
inside addresses, 27
inside interface, 123
inside/outside interface, Telnet and, 314, 315
integrity, 3, 13
Intel Internet Phone, 161
interface command, 77
interface configuration commands, 424
internal threats, 6, 7
Internet filtering. *See* filtering web traffic
Internet Group Management Protocol (IGMP), 203, 205
Internet Key Exchange (IKE), 340–343
  configuring site-to-site IPsec, 349–352
  enabling, 352
  troubleshooting, 369, 591–594
Internet Locator Service (ILS), application inspection for, 164
Internet Phone (Intel), 161
Internet protocol (IP), 50
Internet Scanner, 16
Internet service providers (ISPs), AAA implementation and, 220
Internet traffic, filtering. *See* filtering web traffic
Internet, security issues and, 5
intrusion detection, 175–182
IP (Internet protocol), 50
ip address command, 77, 210
ip address outside dhcp commands, 183
IP addresses
  failover and, 414
  Telnet and, 315
  troubleshooting, 571–573
ip audit commands, 179–181
ip local pool command, 391
IP multicasting, 202
IP Phones (Cisco)
  phone placed on hold and, 164
  SSCP protocol for, 161
ip verify reverse-path commands, 194
  deleting, 197
ip verify statistics command, 196
  deleting, 197
IPsec (Security Architecture for IP), 56, 333–352
  allowing traffic, 350
  design goals of, 335
  troubleshooting, 369, 588–597
isakmp commands, 340, 352
ISAKMP protection suites, 352–354
  pre-shared key for, 354
ISN (initial sequence number), 54
ISPs (Internet service providers), AAA implementation and, 220
ISS Internet Scanner, 16

J

Java applets
  filtering, 173–175
  web filtering and, 166

K

kill command, Telnet and, 315
Kiwi Syslog Daemon, 297

L

L0phtCrack, 16
L2TP (Layer 2 Tunneling Protocol), 383–390
LANs (local area networks), 5
  failover and, 433–443
    disabling, 443
Layer 2 Tunneling Protocol (L2TP), 383–390
LDAP (Lightweight Directory Access Protocol), application inspection for, 164

licensing
for PIX firewalls, 65–71
types of, 67
Lightweight Directory Access Protocol (LDAP), application inspection for, 164
Linux/UNIX systems
FragGuard and, 190
logging and, 299
syslog server software for, 297
load balancing, failover and, 415
local addresses, 27
local area networks (LANs), 5
failover and, 433–443
disabling, 443
local message logging, 290, 291–293
enabling/disabling, 291
local time. *See* date and time
local0...local7, 302
logging, 4
configuring, 290–304
default port for, changing, 295
facilities for, 302
levels of, 291, 299–301
local vs. remote, 290
messages for, capturing/saving via syslog, 293–299
logging buffered command, 292
logging command, logging levels and, 299–301
logging console command, 293
logging history command, 321
logging host command, 294
logging monitor command, 293
logging on command, 291, 321
logging queue command, 296
logging standby command, 296
logging timestamp command, 296
logging trap command, 294

M

MAC addresses, failover and, 414
Mail Guard, 148–150
man tftpd command, 67
managing configurations, 79–82
manual failover, for maintenance, 443
manual IPsec, 369–372
MD5 authentication algorithm, 347
MD5 hashes, 69
RIP protocol and, 201
Media Player (Microsoft), 153, 156
MeetingPoint, 161
memory, performance monitoring for, 609–611
message queue, configuring, 296
messages
logs for. *See* local message logging
stop from printing to console, 293
MIBs (Cisco PIX Management Information Bases), 317
Microsoft
ILS protocol, 164
Media Player, 153, 156
NetMeeting, 161
SMTP implementation and, caution with, 149
Microsoft Exchange servers, caution with SMTP connections, 149
Microsoft Windows systems, syslog server software for, 297
MRTG (Multi Router Traffic Grapher), 316
MS-CHAP (MicrosoftChallenge Handshake Authentication Protocol), 209, 210
Multi Router Traffic Grapher (MRTG), 316
multicast groups, 202
multicast interface command, 204–208
multicasting, 202
debug commands for, 206

N

N2H2 filtering server, 167–169
nameif command, 76
NAS (network access server), 220
nat 0, 165
NAT bypass, 97–100
nat command, 93, 192
NAT. *See* Network Address Translation
National Institute of Standards and Technology (NIST), 14
Nessus, 16
NetMeeting (Microsoft), 161
ILS protocol for, 164
NetRanger appliance, 175
NetShow, 153–157
network access server (NAS), 220
Network Address Translation (NAT), 27, 57
bypassing, 365
troubleshooting, 580–583
Network File System (NFS), 152
Network Information System (NIS), 152
network interfaces
configuring, 76–78
performance monitoring for, 611
network object groups, 119
Network Time Protocol (NTP), 321
authenticating, 325
configuring/verifying, 324
viewing or deleting configuration, 325
Network Translations Inc., 57
NFS (Network File System), 152
NiftyTelnet (Macintosh), 305
NIS (Network Information System), 152
NIST (National Institute of Standards and Technology), 14

no fixup protocol ftp command, 144
nonrepudiation, 4
ntp authenticate command, 325
NTP clients/servers, 324
ntp server command, 324, 326
NTP. *See* Network Time Protocol

O

object grouping, 117–122
object identifiers (OIDs), 317, 318
one-armed routing, 199
Open Systems Interconnect (OSI), 19
OpenSSH (Linux/UNIX), 305
OSI (Open Systems Interconnect), 19
outbound command, 109–111
  case study and, 127–129
outbound connections, 93
outbound traffic, 92–111
  blocking, 100–111
Outlook clients, caution with SMTP connections, 149
outside addresses, 27
outside interface, 123

P

packet filters, 20
  stateful inspection and, 21
packets, fragmented, 189–191
PAP (Password Authentication Protocol), 209, 210
passive mode, 142–146
passphrases, 310
Password Authentication Protocol (PAP), 209, 210
password recovery, 69–71
passwords, 13
  for authenticating users, 4
  Cisco default, 70
  for accessing PIX firewalls, 78
PAT. *See* Port Address Translation
patches, 14
PDM. *See* PIX Device Manager
PentaSafe VigilEnt Security Manager, 16
people hacking, 7
performance
  monitoring, 553–618
  troubleshooting, 602–612
perimeter defense, 7
perimeter security policies, 11
permit statement, IPsec and, 345
PFSS (PIX Firewall Syslog Server), 297
ping timeout, DHCP functionality and, 187
PIX clients, sample configuration for, 397–406
pix default username, 310
PIX Device Manager (PDM), 316, 449–551
  enabling/disabling, 459
  installing/configuring, 455–459
  launching, 460–466
  PIX firewall, configuring/monitoring via, 466–546
  session of, monitoring/ disconnecting, 547
PIX Firewall Syslog Server (PFSS), 297
PIX firewalls, 44–59
  advanced protocol handling by, 55
  case study illustrating configuration for, 122–129
  configuring via PDM, 466–532
  hardware for, 59–65
    PIX models, 59–62
  historical background of, 57
  licensing/upgrades for, 65–71
  monitoring via PDM, 532–546
  number of SSH sessions on, 313
  obtaining SNMP information from, 317
  polling, 317, 318–320
  removing SSH configuration statements, 314
  resetting, 82
  See also firewalls
PKI (Public Key Infrastructure), 321
Point-to-Point Protocol over Ethernet (PPPoE), 209–211
Point-to-Point Tunneling Protocol (PPTP), 372–383
poll keyword, 320
polling PIX firewalls, 317
configuring, 318–320
Port Address Translation (PAT), 29, 58
  troubleshooting, 580–583
port redirection, 115
portmappers, 152
PPP authentication, 209
PPPoE (Point-to-Point Protocol over Ethernet), 209–211
PPTP (Point-to-Point Tunneling Protocol), 372–383
privilege command, 251
protocol object groups, 119
protocols
  advanced, handling, 136–165
  guidelines for selecting, 227
proxy-ARP feature, 198, 199
PRTP (RTP Control Protocol), 155
ps command, 299
public-key cryptography, 4, 346
Public Key Infrastructure (PKI), 321
public SNMP strings, 316
public strings, 318

Q

querying PIX firewalls. *See* polling PIX firewalls
question mark key (?), 74
QuickTime, 153, 155

R

r-utilities, 150–152
Raptor (Symantec), 23
RDT (Real Data Transport), 155
Real-Time Transport Protocol (RTP), 155
RealPlayer, 153
  TCP setting for, 156
Real-Time Streaming Protocol (RTSP), 153–157
reconnaissance attacks, 8
records, accounting and, 223
redundancy, failover and, 415
reload command, 82
remote access
  configuring, 304–316
  tools for managing, 304
Remote Access Dial In User Service (RADIUS), 223–225
  vs. TACACS+, 227
remote message logging, 290
remote procedure call (RPC), 152
remote shell (rsh), 150–152
reset action, 179
resetting the system, 82
resources for further information
  Cisco Secure Policy Manager, 450
  RADIUS and TACACS+ compared, 228
  security policies, 11
  syslog message numbers/descriptions, 571
restricted licensing, 67
reverse-path forwarding (RPF), 194–197
rip command, 200
RIP. *See* Routing Information Protocol
route command, 78, 197
routing, troubleshooting for, 573–580
Routing Information Protocol (RIP), 199–202
  authentication for, 201
RPC (remote procedure call), 152
RPF (reverse-path forwarding), 194–197
RSA key pair, creating, 356
RSA keys
  generating, 306
  removing, 307
rsh (remote shell), 150–152
RTP (Real-Time Transport Protocol), 155
RTP Control Protocol (PRTP), 155
RTP/RTCP connections, 155
RTSP (Real-Time Streaming Protocol), 153–157

S

SARA (Security Auditor's Research Assistant), 16
SAs (security associations), 343–347
script kiddies, 6
SDP (Session Description Protocol), 154
Secure Computing, 23
Secure Corp. case study, 122–129
Secure Shell (SSH), 305–314, 316
  clients for, 305
  vs. remote shell, 150
  removing configuration statements from Cisco PIX, 314
  vs. Telnet, 314
  troubleshooting, 311–314
  verifying configuration for, 308
security, 2–17
  importance of, 2–8
  levels of, 49
  testing, 15–17
Security Architecture for IP. *See* IPsec
security associations (SAs), 343–347
Security Auditor's Research Assistant (SARA), 16
security experts, 16
security policies
  creating, 8–11
  security framework and, 466
security protocols, guidelines for selecting, 227
Security Wheel, 11–17
service object groups, 120–122
service-level agreement (SLA), 414
Session Description Protocol (SDP), 154
Session Initiation Protocol (SIP), 162–164
SHA-1 authentication algorithm, 347
  show commands, performance and, 605–612
show sysopt command, 190
shun command, 182
signatures, 175–179
  disabling, 181
  informational vs. attacks, 178
Simple Mail Transfer Protocol (SMTP), 148–150
Simple Network Management Protocol (SNMP), configuring, 316–321
  managers for, 316
SIP (Session Initiation Protocol), 162–164
SiteServer, ILS protocol for, 164
site-to-site IPsec, 349–352
  without IKE, 369–372
SLA (service-level agreement), 414
SMR configurations, 204–207
SMTP (Simple Mail Transfer Protocol), 148–150
SMTP proxy, 56
SNMP (Simple Network Management Protocol), configuring, 316–321
  managers for, 316
snmp-server commands, 317–320
SNMPc, 316
social engineering, 7
SOHO configuration, for DHCP, 187
SolarWinds, 316
source address, 194

source address spoofing, 21
SQL*Net, 157
SSCP (Skinny Client Control Protocol)
  application inspection for, 161
  phone placed hold and, 164
ssh disconnect command, 313
SSH sessions
  ending, 313
  log message sent to, enabling/ disabling, 293
  number of on Cisco PIX, 313
  timeout for, 307
ssh timeout command, 307
SSH. *See* Secure Shell
ssh_known_hosts file, configuring (Tera Term), 309
state, 47–49
stateful failover, 420
stateful inspection, 21
static address translation, 27, 112
static command, 112, 115, 192
  crypto access lists and, 364
static routes, 78, 197–199
Stratum 1/Stratum 2 NTP servers, 324
streaming, 153–157
structured attacks, 6
stub multicast routing, 202–209
summer time, 322
Sun Microsystems RPC, 152
support options as tools for troubleshooting, 601
Symantec Enterprise Security Manager (ESM), 16
Symantec Raptor, 23
symmetric encryption, 346
SYN floodguard, 192–194
SYN Received connections, 192
syslog, 290, 293–299
  blocked network traffic and, 295
  server software for, 297
syslog message queue, 296
syslog messages
  disabling selected, 303
  ID numbers for, 303
  plain vs. encrypted, 296
  searching for intrusion detection support, 175
syslogd daemon, 299
sysopt connection permit commands, 350, 387
sysopt connection permit pptp command, 375
sysopt connection permit-ipsec command, 351
sysopt noproxyarp inside_interface command, 147
sysopt security fragguard command, 190
system date and time. *See* date and time
system identification, configuring for SNMP, 317
system management, configuring, 289–331
system properties, configuring, 474–500

T

TACACS+ (Terminal Access Controller Access Control System Plus), 225–228
  vs. RADIUS, 227
TCP (Transmission Control Protocol), 51–54
TCP Intercept, 193
TCP syslog, 295
  blocked network traffic and, 295
TCP/IP communication, SSH for, 305–314
Telnet, 314
telnet command, 314
Telnet sessions, enabling/ disabling log messages sent to, 293
Tera Term (Windows), 305
  configuring for SSH sessions, 308
    for automatic use, 310
Terminal Access Controller Access Control System Plus (TACACS+), 225–228
  vs. RADIUS, 227
terminal logging, 293
terminal monitor command, 293
TFTP servers, 67
  installing via PDM, 457
threats, 6
time/time zone. *See* date and time
timeout
  for datastreams, 160
  for SIP, 163
  for SSH sessions, 307
  for Telnet sessions, 314
timeout h323 command, 160
timeout sip command, 163
timestamps, 296, 300
  consistency of, 321
tools for security testing, 16
traffic, 92–133
  capturing, 597–602
    downloads of, 600
  IPsec, allowing, 350
  specifying all, 98
transaction-logging systems, 3
transform sets, defining, 364
translation rules, 505–511
translation tables (xlate tables), 48, 93, 137
Transmission Control Protocol (TCP), 51–54
transport mode (IPsec), 338
trap keyword, 320
traps, 317
  configuring/using, 320
  stopping, 321
troubleshooting, 553–618
  Internet Key Exchange, 369
  performance, 602–612

support options as tools for, 601
tunnel mode (IPsec), 338
TurboACLs, 116

U

uauth timer, 262
UDP (User Datagram Protocol), 54, 138
UDP syslog, 295
unicast routing, 197–202
UNIX syslogging, 299
UNIX systems. *See* Linux/UNIX systems
unprivileged mode, 72
unrestricted licensing, 67
unstructured threats, 6
upgrades for PIX firewalls, 65–71
uptime requirement, 414
url-block block command, 171
url-block url-size command, 170
url-block url-mempool command, 172
url-cache command, 170
URL filtering, 165–172
  long URLs and, 169
  support for, 57
URL logging, 301
url-server command, 167, 168
User Datagram Protocol (UDP), 54, 138
username command, 243, 250
usernames
  pix default, 310
  troubleshooting incorrect, 312
UTC (Coordinated Universal Time), 321

V

VDO Live, 153–157
vendors of firewalls, 18
VigilEnt Security Manager, 16
virtual http command, 266–268
virtual private networks (VPNs), 29–31
  configuring, 333–411
    via PDM, 519–532
  connection graphs for, 539
  support for, 56
virtual telnet command, 268–270
Voice over IP (VoIP), 159–164
vpdn commands, 209
vpdn username command, 376
VPN clients, sample configuration for, 397–406
VPNs. *See* virtual private networks

W

WANs (wide area networks), 5
web filtering. *See* filtering web traffic
web sites
  Castle Rock, 318
  Cisco
    MIBs, 317
    OIDs, 318
    TFTP servers, 68
  IDS signatures, 178
  Kiwi Syslog Daemon, 297
  MRTG, 317
  N2H2 filtering server, 167
  patch sources, 14
  PIX Firewall Syslog Server, 297
  Somix, 317
  syslog message ID numbers, 303
  vendors of security testing tools, 16
  Websense filtering server, 167
  Zip World, 308
web traffic, filtering. *See* filtering web traffic
Websense filtering server, 167–169
whisker, 16
who command, Telnet and, 315
wide area network (WANs), 5
Windows 2000 VPN clients
  Layer 2 tunneling and, 389
  point-to-point tunneling and, 380–383
WINS servers, configuring for DHCP functionality, 186
write command, 79
write memory command, 308, 418
write standby command, 418

X

XDR format, 152
xlate tables. *See* translation tables

Z

zero-day exploits, 6

Ultimate CD-ROM Collections Now Available!

Incredible Value for Only $99.99 each

Receive the CD of your choice with seven books in PDF format.
You would pay over $350 for these books in hard copy format.

Ultimate Cisco CD

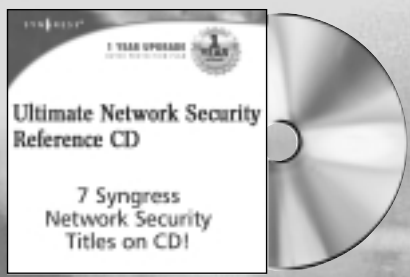
Managing Cisco Network Security (List Price: $59.95)
Building Cisco Networks for Windows 2000 (List Price: $59.95)
Configuring Cisco AVVID (List Price: $59.95)
Administering Cisco QoS for IP Networks (List Price: $59.95)
Building Cisco Remote Access Networks (List Price: $49.95)
Configuring Cisco Voice Over IP (List Price: $59.95)
Cisco AVVID and IP Telephony Design & Implementation (List Price: $69.95)

Ultimate Windows 2000 CD

Managing Active Directory for Windows 2000 Server (List Price: $49.95)
Managing Windows 2000 Network Services (List Price: $49.95)
Configuring Windows 2000 Server Security (List Price: $49.95)
Windows 2000 Server System Administration Handbook (List Price: $49.95)
Deploying Windows 2000 with Support Tools (List Price: $49.95)
Windows 2000 Configuration Wizards (List Price: $39.95)
Troubleshooting Windows 2000 TCP/IP (List Price: $49.95)

Ultimate .NET CD

BizTalk Server 2000 Developer's Guide for .NET (List Price: $49.95)
XML .NET Developer's Guide (List Price: $49.95)
VB .NET Developer's Guide (List Price: $49.95)
ASP .NET Web Developer's Guide (List Price: $49.95)
C# .NET Web Developer's Guide (List Price: $49.95)
.NET Mobile Web Developer's Guide (List Price: $49.95)
Designing SQL Server 2000 Databases for .NET Enterprise Servers (List Price: $49.95)



Ultimate Network Security CD

Hack Proofing Your Network (List Price: $49.95)
Hack Proofing Windows 2000 Server (List Price: $49.95)
Hack Proofing Linux: A Guide to Open Source Security (List Price: $49.95)
Hack Proofing Sun Solaris (List Price: $49.95)
Hack Proofing Your E-Commerce Site (List Price: $49.95)
Hack Proofing Your Web Applications (List Price $49.95)
Managing Cisco Network Security: Building Rock Solid Networks (List Price $59.95)

Visit www.syngress.com for details.

SYNGRESS®