

Application Guide



Alteon OS™ 20.1 Alteon Application Switch

Part Number: 315394-B, January 2003

NORTEL
NETWORKS™

4655 Great America Parkway
Santa Clara, CA 95054
Phone 1-800-4Nortel
www.nortelnetworks.com

Copyright 2003 Nortel Networks, Inc., 4655 Great America Parkway, Santa Clara, California 95054, USA.
All rights reserved. Part Number: 315394-B.

This document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this document may be reproduced in any form by any means without prior written authorization of Nortel Networks, Inc. Documentation is provided “as is” without warranty of any kind, either express or implied, including any kind of implied or express warranty of non-infringement or the implied warranties of merchantability or fitness for a particular purpose.

U.S. Government End Users: This document is provided with a “commercial item” as defined by FAR 2.101 (Oct 1995) and contains “commercial technical data” and “commercial software documentation” as those terms are used in FAR 12.211-12.212 (Oct 1995). Government End Users are authorized to use this documentation only in accordance with those rights and restrictions set forth herein, consistent with FAR 12.211-12.212 (Oct 1995), DFARS 227.7202 (JUN 1995) and DFARS 252.227-7015 (Nov 1995).

Nortel Networks, Inc. reserves the right to change any products described herein at any time, and without notice. Nortel Networks, Inc. assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by Nortel Networks, Inc. The use and purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of Nortel Networks, Inc.

Alteon OS, Alteon 2424, Alteon 2224, Alteon 180, Alteon 180e, Alteon 184, Alteon AD3, Alteon AD4, and ACEswitch are trademarks of Nortel Networks, Inc. in the United States and certain other countries. Cisco® and EtherChannel® are registered trademarks of Cisco Systems, Inc. in the United States and certain other countries. Check Point® and FireWall-1® are trademarks or registered trademarks of Check Point Software Technologies Ltd. Any other trademarks appearing in this manual are owned by their respective companies.

Originated in the U.S.A.



Contents

Preface 19

Who Should Use This Guide 19

What You'll Find in This Guide 19

Typographic Conventions 21

How to Get Help 22

Part 1: Basic Switching 23

Chapter 1: Accessing the Switch 25

Using the CLI 26

Using Alteon EMS 27

Using the Browser-Based Interface 29

Using SNMP 29

Using the Management Port 30

 Setting Up the Management Port 31

Securing the Switch 32

 Setting Allowable Source IP Address Ranges 33

 RADIUS Authentication and Authorization 34

 Secure Shell and Secure Copy 37

Chapter 2: VLANs 43

VLAN ID Numbers 44

VLAN Tagging 44

VLANs and the IP Interfaces 45

VLAN Topologies and Design Issues 45

 Example 1: Multiple VLANs with Tagging Adapters 46

 Example 2: Parallel Links with VLANs 48

VLANs and Default Gateways	49
Segregating VLAN Traffic	49
Configuring the Local Network	51
Configuring Gateways per VLAN	51

Chapter 3: Port Trunking 55

Overview	55
Statistical Load Distribution	56
Built-In Fault Tolerance	56
Port Trunking Example	57

Chapter 4: Spanning Tree Protocol 61

Overview	62
Bridge Protocol Data Units (BPDUs)	63
Determining the Path for Forwarding BPDUs	63
Multiple Spanning Trees	64
Why Do We Need Multiple Spanning Trees?	64
Four-Switch Topology with a Single Spanning Tree	64
Four-Switch Topology with Multiple Spanning Trees	65
Switch-Centric Spanning Tree Protocol	67
VLAN Participation in Spanning Tree Groups	67
Configuring Multiple Spanning Tree Groups	68

Part 2: IP Routing 71

Chapter 5: Basic IP Routing 73

IP Routing Benefits	74
Routing Between IP Subnets	74
Example of Subnet Routing	77
Defining IP Address Ranges for the Local Route Cache	81
Dynamic Host Configuration Protocol	82
DHCP Relay Agent	83
DHCP Relay Agent Configuration	84

Chapter 6: Routing Information Protocol 85

Distance Vector Protocol	85
Stability	85
Routing Updates	86

Chapter 7: Border Gateway Protocol 87

Internal Routing Versus External Routing	88
Forming BGP Peer Routers	89
What is a Route Map?	89
Incoming and Outgoing Route Maps	90
Precedence	91
Configuration Overview	91
Aggregating Routes	93
Redistributing Routes	93
BGP Attributes	94
Local Preference Attribute	94
Metric (Multi-Exit Discriminator) Attribute	94
Selecting Route Paths in BGP	95
BGP Failover Configuration	96
Default Redistribution and Route Aggregation Example	99

Chapter 8: OSPF 101

OSPF Overview	102
Types of OSPF Areas	102
Types of OSPF Routing Devices	104
Neighbors and Adjacencies	105
The Link-State Database	105
The Shortest Path First Tree	106
Internal Versus External Routing	106
OSPF Implementation in Alteon OS	107
Configurable Parameters	107
Defining Areas	108
Interface Cost	110
Electing the Designated Router and Backup	110
Summarizing Routes	110
Default Routes	111
Virtual Links	112
Router ID	113
Authentication	113
Host Routes for Load Balancing	116
OSPF Features Not Supported in This Release	116

OSPF Configuration Examples	117
Example 1: Simple OSPF Domain	118
Example 2: Virtual Links	120
Example 3: Summarizing Routes	124
Example 4: Host Routes	127
Verifying OSPF Configuration	133

Part 3: Application Switching Fundamentals 135

Chapter 9: Server Load Balancing 137

Understanding Server Load Balancing	138
Identifying Your Network Needs	138
How Server Load Balancing Works	139
Implementing Basic Server Load Balancing	141
Network Topology Requirements	142
Configuring Server Load Balancing	144
Additional Server Load Balancing Options	148
Extending SLB Topologies	156
Virtual Matrix Architecture	156
Proxy IP Addresses	156
Mapping Ports	160
Direct Server Interaction	163
Delayed Binding	167
Load Balancing Special Services	170
IP Server Load Balancing	170
FTP Server Load Balancing	171
Domain Name Server (DNS) Load Balancing	172
Real Time Streaming Protocol SLB	176
Wireless Application Protocol SLB	181
Intrusion Detection System Server Load Balancing	188
WAN Link Load Balancing	198

Chapter 10: Filtering 205

Overview	206
Filtering Benefits	206
Filtering Criteria	206
Stacking Filters	207
Overlapping Filters	208
The Default Filter	209
VLAN-based Filtering	210
Optimizing Filter Performance	212
Filter Logs	212
IP Address Ranges	214
Cache-Enabled versus Cache-Disabled Filters	214
TCP Rate Limiting	215
Configuring TCP Rate Limiting Filters	216
Tunable Hash for Filter Redirection	220
Filter-based Security	221
Network Address Translation	227
Static NAT	227
Dynamic NAT	229
FTP Client NAT	231
Matching TCP Flags	233
Matching ICMP Message Types	238

Chapter 11: Application Redirection 241

Overview	242
Cache Redirection Environment	242
Additional Application Redirection Options	243
RTSP Cache Redirection	249
IP Proxy Addresses for NAT	252
Excluding Noncacheable Sites	254

Chapter 12: Health Checking 255

Real Server Health Checks	257
DSR Health Checks	258
Link Health Checks	259
Configuring the Switch for Link Health Checks	259
TCP Health Checks	260
ICMP Health Checks	260

Script-Based Health Checks	261
Configuring the Switch for Script-Based Health Checks	261
Script Format	262
Scripting Guidelines	263
Script Configuration Examples	263
Application-Specific Health Checks	266
HTTP Health Checks	267
UDP-Based DNS Health Checks	269
SNMP Health Check	270
FTP Server Health Checks	271
POP3 Server Health Checks	272
SMTP Server Health Checks	273
IMAP Server Health Checks	274
NNTP Server Health Checks	275
RADIUS Server Health Checks	276
HTTPS/SSL Server Health Checks	277
WAP Gateway Health Checks	277
LDAP Health Checks	280
ARP Health Checks	282
Failure Types	283
Service Failure	283
Server Failure	283

Chapter 13: High Availability 285

VRRP Overview	286
VRRP Components	286
VRRP Operation	289
Selecting the Master VRRP Router	289
Active-Standby Failover	290
Failover Methods	292
Active-Standby Redundancy	293
Active-Active Redundancy	294
Hot-Standby Redundancy	295
Synchronizing Configurations	297
Alteon OS Extensions to VRRP	298
Virtual Server Routers	298
Sharing and Active-Active Failover	299
Tracking VRRP Router Priority	300

High Availability Configurations	302
Active-Standby Virtual Server Router Configuration	302
Active-Active VIR and VSR Configuration	304
Active-Active Server Load Balancing Configuration	306
VRRP-Based Hot-Standby Configuration	314
Virtual Router Deployment Considerations	316
Mixing Active-Standby and Active-Active Virtual Routers	316
Synchronizing Active/Active Failover	316
Eliminating Loops with STP and VLANs	317
Assigning VRRP Virtual Router ID	319
Configuring the Switch for Tracking	319
Synchronizing Configurations	321
Stateful Failover of Layer 4 and Layer 7 Persistent Sessions	322
What Happens When a Switch Fails	323
Viewing Statistics on Persistent Port Sessions	325

Part 4: Advanced Switching 327

Chapter 14: Global Server Load Balancing 329

GSLB Overview	330
Benefits	330
How GSLB Works	331
Enabling GSLB on the Switch	332
Configuring GSLB	333
Proxy IP for Non-HTTP Redirects	344
How Proxy IP Works	346
Configuring Proxy IP Addresses	347
Verifying GSLB Operation	348
Configuring GSLB Client Proximity Tables	348
Using Border Gateway Protocol for GSLB	351

Chapter 15: Firewall Load Balancing 353

Firewall Overview	354
Basic FWLB	356
Basic FWLB Implementation	357
Configuring Basic FWLB	359

Four-Subnet FWLB	368
Four-Subnet FWLB Implementation	370
Configuring Four-Subnet FWLB	371
Advanced FWLB Concepts	389
Free-Metric FWLB	389
Adding a Demilitarized Zone (DMZ)	393
Firewall Health Checks	395

Chapter 16: Virtual Private Network Load Balancing 397

Overview	398
How VPN Load Balancing Works	398
VPN Load-Balancing Configuration	400
Configure the First Clean-Side Application Switch-CA	401
Configure the Second Clean-Side Application Switch-CB	404
Configure the First Dirty-Side Application Switch-DA	406
Configure the Second Dirty-Side Application Switch-DB	409
Test Configurations and General Topology	412
Test the VPN	413

Chapter 17: Content Intelligent Switching 415

Overview	416
Parsing Content	417
HTTP Header Inspection	418
Buffering Content with Multiple Frames	418
Content Intelligent Server Load Balancing	419
URL-Based Server Load Balancing	419
Virtual Hosting	424
Cookie-Based Preferential Load Balancing	427
Browser-Smart Load Balancing	430
URL Hashing for Server Load Balancing	431
Header Hash Load Balancing	433
DNS Load Balancing	434
RTSP Load Balancing	437

Content Intelligent Cache Redirection	442
URL-Based Cache Redirection	443
HTTP Header-Based Cache Redirection	451
Browser-Based Cache Redirection	452
URL Hashing for Cache Redirection	453
RTSP Streaming Cache Redirection	456
Exclusionary String Matching for Real Servers	461
Configuring for Exclusionary URL String Matching	461
Regular Expression Matching	463
Standard Regular Expression Characters	463
Configuring Regular Expressions	464
Content Precedence Lookup	465
Requirements	466
Using the <i>or</i> and <i>and</i> Operators	466
Assigning Multiple Strings	467
Layer 7 Deny Filters	468

Chapter 18: Persistence 471

Overview of Persistence	472
Using Source IP Address	472
Using Cookies	473
Using SSL Session ID	473
Cookie-Based Persistence	474
Permanent and Temporary Cookies	475
Cookie Formats	475
Cookie Properties	476
Client Browsers that Do Not Accept Cookies	476
Cookie Modes of Operation	477
Configuring Cookie-Based Persistence	480
Server-Side Multi-Response Cookie Search	486
SSL Session ID-Based Persistence	487
How SSL Session ID-Based Persistence Works	487

Chapter 19: Bandwidth Management 491

- Enabling BWM on the Switch 492
- Bandwidth Management Contracts 493
- Bandwidth Policies 495
 - Rate Limits 496
 - Bandwidth Policy Configuration 496
- Data Pacing for Traffic Shaping Contracts 497
- Rate Limiting Timeslots 498
- Classification Criteria 498
 - Server Output Bandwidth Control 498
 - Application Bandwidth Control 499
 - Combinations 499
 - Precedence 499
 - URL-Based Bandwidth Management 500
 - HTTP Header-Based Bandwidth Management 502
 - Cookie-Based Bandwidth Management 503
- Bandwidth Statistics and History 504
 - Statistics Maintained 504
 - Statistics and Management Information Bases 504
- Packet Coloring (TOS bits) for Burst Limit 505
- Configuring Bandwidth Management 506
 - Additional Configuration Examples 509
 - Preferential Services Examples 512

Appendix A: Troubleshooting 523

- Monitoring Ports 524
- Filtering the Session Dump 526

Glossary 527

Index 531



Figures

Figure 1-1:	RADIUS Authentication and Authorization: How It Works	34
Figure 2-1:	Example 1: Multiple VLANs with VLAN-Tagged Gigabit Adapters	46
Figure 2-2:	Example 2: Parallel Links with VLANs	48
Figure 2-3:	Default Gateways per VLAN	49
Figure 3-1:	Port Trunk Group	55
Figure 3-2:	Port Trunk Group Configuration Example	57
Figure 4-1:	Using Multiple Instances of Spanning Tree Protocol	64
Figure 4-2:	VLAN 3 Isolated in a Single Spanning Tree Group	65
Figure 4-3:	Implementing Multiple Spanning Tree Groups	66
Figure 5-1:	The Router Legacy Network	75
Figure 5-2:	Switch-Based Routing Topology	76
Figure 5-3:	DHCP Relay Agent Configuration	84
Figure 7-1:	iBGP and eBGP	88
Figure 7-2:	Distributing Network Filters in Access Lists and Route Maps	90
Figure 7-3:	BGP Failover Configuration Example	96
Figure 7-4:	Route Aggregation and Default Route Redistribution	99
Figure 8-1:	OSPF Area Types	103
Figure 8-2:	OSPF Domain and an Autonomous System	104
Figure 8-3:	Injecting Default Routes	111
Figure 8-4:	OSPF Authentication	113
Figure 8-5:	A Simple OSPF Domain	118
Figure 8-6:	Configuring a Virtual Link	120
Figure 8-7:	Summarizing Routes	124
Figure 8-8:	Configuring OSPF Host Routes	127
Figure 9-1:	Traditional Versus SLB Network Configurations	139
Figure 9-2:	Web Hosting Configuration Without SLB	141
Figure 9-3:	Web Hosting with SLB Solutions	141
Figure 9-4:	SLB Client/Server Traffic Routing	142
Figure 9-5:	Example Network for Client/Server Port Configuration	143
Figure 9-6:	Basic Virtual Port to Real Port Mapping Configuration	161
Figure 9-7:	Direct Server Return	164
Figure 9-8:	Mapped and Nonmapped Server Access	165
Figure 9-9:	DoS SYN Attacks without Delayed Binding	167

Figure 9-10:	Repelling DoS SYN Attacks With Delayed Binding	168
Figure 9-11:	Layer 4 DNS Load Balancing	172
Figure 9-12:	Load Balancing RTSP Servers	178
Figure 9-13:	Load Balancing WAP Gateways	182
Figure 9-14:	Server load balancing and IDS Load balancing	190
Figure 9-15:	Server load balancing and IDS load balancing	194
Figure 9-16:	WAN Link Load Balancing	200
Figure 10-1:	Assigning Filters According to Range of Coverage	208
Figure 10-2:	Assigning Filters to Overlapping Ranges	208
Figure 10-3:	Assigning a Default Filter	209
Figure 10-4:	VLAN-based Filtering	210
Figure 10-5:	Configuring Clients with Different Rates	216
Figure 10-6:	Limiting User Access to Server	219
Figure 10-7:	Security Topology Example	221
Figure 10-8:	Static Network Address Translation	228
Figure 10-9:	Dynamic Network Address Translation	229
Figure 10-10:	Active FTP for Dynamic NAT	231
Figure 10-11:	TCP ACK Matching Network	233
Figure 11-1:	Traditional Network Without Cache Redirection	242
Figure 11-2:	Network with Cache Redirection	243
Figure 11-3:	RTSP Cache Redirection	249
Figure 13-1:	Example 1: VRRP Router	288
Figure 13-2:	Example 2: VRRP Router	291
Figure 13-3:	A Non-VRRP, Hot-Standby Configuration	292
Figure 13-4:	Active-Standby Redundancy	293
Figure 13-5:	Active-Active Redundancy	294
Figure 13-6:	Hot-Standby Redundancy	295
Figure 13-7:	Active-Active High Availability	299
Figure 13-8:	Active-Standby High-Availability Configuration	302
Figure 13-9:	Active-Active High-Availability Configuration	304
Figure 13-10:	Hot-Standby Configuration	314
Figure 13-11:	Loops in Active-Active Configuration	317
Figure 13-12:	Cross-Redundancy Creates Loops, but STP Resolves Them	318
Figure 13-13:	Using VLANs to Create Non-Looping Topologies	318
Figure 13-14:	Stateful Failover Example when the Master Switch Fails	323
Figure 14-1:	DNS Resolution with Global Server Load Balancing	331
Figure 14-2:	GSLB Topology Example	334
Figure 14-3:	HTTP and Non-HTTP Redirects	345
Figure 14-4:	POP3 Request Fulfilled via IP Proxy	346
Figure 14-5:	GSLB Proximity Tables: How They Work	349
Figure 14-6:	Configuring Client Proximity Table	350

Figure 15-1:	Typical Firewall Configuration Before FWLB	354
Figure 15-2:	Basic FWLB Topology	356
Figure 15-3:	Basic FWLB Process	357
Figure 15-4:	Basic FWLB Example Network	359
Figure 15-5:	Four-Subnet FWLB Topology	368
Figure 15-6:	Four-Subnet FWLB Process	370
Figure 15-7:	Four-Subnet FWLB Example Network	371
Figure 15-8:	Basic FWLB Example Network	389
Figure 15-9:	Four-Subnet FWLB Example Network	391
Figure 15-10:	Typical Firewall Load-Balancing Topology with DMZ	393
Figure 16-1:	Basic Network Frame Flow and Operation	399
Figure 16-2:	VPN Load-Balancing Configuration Example	400
Figure 16-3:	Checkpoint Rules for Both VPN Devices as Seen in the Policy Editor	412
Figure 17-1:	Content Intelligent Load Balancing Example	416
Figure 17-2:	URL-Based Server Load Balancing	420
Figure 17-3:	Using URL hashing to Load Balance Nontransparent Caches	431
Figure 17-4:	Load Balancing DNS Queries	434
Figure 17-5:	Layer 7 RTSP Load Balancing	438
Figure 17-6:	URL-Based Cache Redirection	444
Figure 17-7:	URL Hashing for Application Redirection	455
Figure 17-8:	Load Balancing RTSP Cache Servers	457
Figure 17-9:	Content Precedence Lookup Protectors Example	466
Figure 17-10:	Content Precedence Lookup Multiple Strings Example	467
Figure 17-11:	Configuring Layer 7 Deny Filter	469
Figure 18-1:	Cookie-Based Persistence: How It Works	474
Figure 18-2:	Insert Cookie Mode	477
Figure 18-3:	Passive Cookie Mode	478
Figure 18-4:	Rewrite Cookie Mode	479
Figure 18-5:	SSL Session ID-Based Persistence	488
Figure 19-1:	Bandwidth Management: How It Works	493
Figure 19-2:	Bandwidth Rate Limits	495
Figure 19-3:	Virtual Clocks and TDT	497
Figure 19-4:	URL-Based Bandwidth Management	501
Figure 19-5:	URL-Based Bandwidth Management with Cache Redirection	502
Figure 19-6:	Cookie-Based Bandwidth Management	503
Figure 19-7:	Cookie-Based Preferential Services	519



Tables

Table 1-1: User Access Levels	36
Table 1-2: Alteon OS-proprietary Attributes for Radius	37
Table 2-1: Route Cache Example	50
Table 4-1: Ports, Trunk Groups, and VLANs	62
Table 4-2: Multiple Spanning Tree Groups per VLAN	67
Table 5-1: Subnet Routing Example: IP Address Assignments	77
Table 5-2: Subnet Routing Example: IP Interface Assignments	77
Table 5-3: Subnet Routing Example: Optional VLAN Ports	79
Table 5-4: Local Routing Cache Address Ranges	81
Table 9-1: Web Host Example: Real Server IP Addresses	144
Table 9-2: Web Host Example: Port Usage	146
Table 9-3: Well-Known Application Ports	148
Table 9-4: Proxy IP addresses on Alteon Application Switch 2424	157
Table 9-5: Proxy IP addresses on Alteon Application Switch 2224	157
Table 9-6: Proxy Example: Port Usage	158
Table 9-7: Setting Up IDS Servers	191
Table 10-1: Well-Known Protocol Types	207
Table 10-2: Filtering IP Address Ranges	214
Table 10-3: Web Cache Example: Real Server IP Addresses	222
Table 10-4: TCP Flags	233
Table 10-5: ICMP Message Types	238
Table 11-1: Cache Redirection Example: Real Server IP Addresses	244
Table 13-1: Active Standby Configuration	291
Table 13-2: Sharing Active-Active Failover	299
Table 13-3: VRRP Tracking Parameters	300
Table 14-1: GSLB Example: California Real Server IP Addresses	336
Table 14-2: GSLB Example: California Alteon Switch Port Usage	337
Table 14-3: Denver Real Server IP Addresses	340
Table 14-4: Web Host Example: Port Usage	341
Table 14-5: HTTP Versus Non-HTTP Redirects	344
Table 17-1: Standard Regular Expression Special Characters	463
Table 17-2: Real Server Content	468
Table 18-1: Comparison Among the Three Cookie Modes	477

Table 19-1: Bandwidth Rate Limits 496



Preface

This *Application Guide* describes how to configure and use the Alteon OS software on the Alteon Application Switches. For documentation on installing the switches physically, see the *Hardware Installation Guide* for your particular switch model.

Who Should Use This Guide

This *Application Guide* is intended for network installers and system administrators engaged in configuring and maintaining a network. The administrator should be familiar with Ethernet concepts, IP addressing, Spanning Tree Protocol, and SNMP configuration parameters.

What You'll Find in This Guide

This guide will help you plan, implement, and administer Alteon OS software. Where possible, each section provides feature overviews, usage examples, and configuration instructions.

Part 1: Basic Switching

- [Chapter 1, “Accessing the Switch,”](#) describes how to access the Alteon Application Switch to configure, view information and run statistics on the switch. This chapter also discusses different methods to manage the switch for remote administrators using specific IP addresses, RADIUS authentication, Secure Shell (SSH), and Secure Copy (SCP).
- [Chapter 2, “VLANs,”](#) describes how to configure Virtual Local Area Networks (VLANs) for creating separate network segments, including how to use VLAN tagging for devices that use multiple VLANs. This chapter also describes how Jumbo frames can be used to ease server processing overhead.
- [Chapter 3, “Port Trunking,”](#) describes how to group multiple physical ports together to aggregate the bandwidth between large-scale network devices.

- [Chapter 4, “Spanning Tree Protocol,”](#) discusses how Spanning Trees configure the network so that the switch uses the most efficient path when multiple paths exist.

Part 2: IP Routing

- [Chapter 5, “Basic IP Routing,”](#) describes how to configure the Alteon Application Switch for IP routing using IP subnets, and DHCP Relay.
- [Chapter 6, “Routing Information Protocol,”](#) describes how the Alteon OS software implements standard RIP for exchanging TCP/IP route information with other routers
- [Chapter 7, “Border Gateway Protocol,”](#) describes BGP concepts and BGP features supported in Alteon OS.
- [Chapter 8, “OSPF,”](#) describes OSPF concepts, how OSPF is implemented in Alteon OS, and four examples of how to configure your switch for OSPF support.

Part 3: Application Switching Fundamentals

- [Chapter 9, “Server Load Balancing,”](#) describes how to configure the Alteon Application Switch to balance network traffic among a pool of available servers for more efficient, robust, and scalable network services.
- [Chapter 10, “Filtering,”](#) describes how to configure and optimize network traffic filters for security and Network Address Translation.
- [Chapter 11, “Application Redirection,”](#) describes how to use filters for redirecting traffic to such network streamlining devices as caches.
- [Chapter 12, “Health Checking,”](#) describes how to configure the Alteon Application Switch to recognize the availability of the various network resources used with the various load-balancing and application redirection features.
- [Chapter 13, “High Availability,”](#) describes how to use the Virtual Router Redundancy Protocol (VRRP) to ensure that network resources remain available if one Alteon Application Switch is removed for service.

Part 4: Advanced Switching

- [Chapter 14, “Global Server Load Balancing,”](#) describes configuring Server Load Balancing across multiple geographic sites.
- [Chapter 15, “Firewall Load Balancing,”](#) describes how to combine features to provide a scalable solution for load balancing multiple firewalls.
- [Chapter 16, “Virtual Private Network Load Balancing,”](#) describes using your Alteon Application Switch to load balance secure point-to-point links.

- Chapter 17, “Content Intelligent Switching,” describes how to perform load balancing and application redirection based on Layer 7 packet content information (such as URL, HTTP Header, browser type, and cookies).
- Chapter 18, “Persistence,” describes how to ensure that all connections from a specific client session reach the same server. Persistence can be based on cookies or SSL session ID.
- Chapter 19, “Bandwidth Management,” describes how to configure the Alteon Application Switch for allocating specific portions of the available bandwidth for specific users or applications.
- Appendix A, “Troubleshooting,” discusses two tools for troubleshooting your application switch—monitoring ports and filtering session dumps.

Typographic Conventions

The following table describes the typographic styles used in this book.

Table 1 Typographic Conventions

Typeface or Symbol	Meaning	Example
AaBbCc123	This type is used for names of commands, files, and directories used within the text. It also depicts on-screen computer output and prompts.	View the <code>readme.txt</code> file. Main#
AaBbCc123	This bold type appears in command examples. It shows text that must be typed in exactly as shown.	Main# sys
<AaBbCc123>	This italicized type appears in command examples as a parameter placeholder. Replace the indicated text with the appropriate real name or value when using the command. Do not type the brackets. This also shows book titles, special terms, or words to be emphasized.	To establish a Telnet session, enter: host# telnet <i><IP address></i> Read your <i>User's Guide</i> thoroughly.
[]	Command items shown inside brackets are optional and can be used or excluded as the situation demands. Do not type the brackets.	host# ls [-a]

How to Get Help

If you purchased a service contract for your Nortel Networks product from a distributor or authorized reseller, contact the technical support staff for that distributor or reseller for assistance.

If you purchased a Nortel Networks service program, contact one of the following Nortel Networks Technical Solutions Centers:

Technical Solutions Center	Telephone
Europe, Middle East, and Africa	00800 8008 9009 or +44 (0) 870 907 9009
North America	(800) 4NORTEL or (800) 466-7835
Asia Pacific	(61) (2) 8870-8800
China	(800) 810-5000

Additional information about the Nortel Networks Technical Solutions Centers is available at the following URL:

<http://www.nortelnetworks.com/help/contact/global>

An Express Routing Code (ERC) is available for many Nortel Networks products and services. When you use an ERC, your call is routed to a technical support person who specializes in supporting that product or service. To locate an ERC for your product or service, refer to the following URL:

<http://www.nortelnetworks.com/help/contact/erc/index.html>

Part 1: Basic Switching

This section discusses basic Layer 1—2 switching functions. This includes how to access and manage the switch:

- Accessing the switch
- VLANs
- Port Trunking
- Spanning Tree Protocol



CHAPTER 1

Accessing the Switch

The Alteon OS software provides means for accessing, configuring, and viewing information and statistics about the Alteon Application Switch. This chapter discusses different methods of accessing the switch and ways to secure the switch for remote administrators:

- “Using the CLI” on page 26
- “Using Alteon EMS” on page 27
- “Using the Browser-Based Interface” on page 29
- “Using SNMP” on page 29
- “Using the Management Port” on page 30
- “Securing the Switch” on page 32
 - “Setting Allowable Source IP Address Ranges” on page 33
 - “RADIUS Authentication and Authorization” on page 34
 - “Secure Shell and Secure Copy” on page 37

Using the CLI

The Command Line Interface (CLI) is a built-in, text-based menu system for access via local terminal or remote Telnet session. The CLI is the most direct method for collecting switch information and performing switch configuration. The Main Menu of the CLI with administrator privileges is displayed in the following table:

[Main Menu]	
info	- Information Menu
stats	- Statistics Menu
cfg	- Configuration Menu
oper	- Operations Command Menu
boot	- Boot Options Menu
maint	- Maintenance Menu
diff	- Show pending config changes [global command]
apply	- Apply pending config changes [global command]
save	- Save updated config to FLASH [global command]
revert	- Revert pending or applied changes [global command]
exit	- Exit [global command, always available]

You can access the built-in, text-based CLI in the following ways:

- Using a serial connection via the console port

You can access and configure the application switch by using a computer running terminal emulation software.

- Using the management port

The management port is a Fast Ethernet port on the Alteon Application Switch that is used exclusively for managing the switch.

For more information on the management port, see [“Using the Management Port” on page 30](#).

- Using a Telnet connection over the network

A Telnet connection offers the convenience of accessing the switch from any workstation connected to the network. Telnet access provides the same options for user and administrator access as those available through the console port.

To establish a Telnet connection with the switch, run the Telnet program on your workstation and issue the Telnet command, followed by the switch IP address:

```
telnet <switch IP address>
```

- Using a SSH connection to securely log into another computer over a network.

The SSH (Secure Shell) protocol enables you to securely log into another computer over a network to execute commands remotely. As a secure alternative to using Telnet to manage switch configuration, SSH ensures that all data sent over the network is encrypted and secure. For more information, see [“Secure Shell and Secure Copy” on page 37](#).

For more information on the CLI, see the *Alteon OS Command Reference*.

Using Alteon EMS

The Alteon Element Management System (EMS) is a Java GUI application that runs on many platforms used for configuring and monitoring Alteon Application Switches. The application communicates with the switches via the SNMP protocol. Alteon EMS can be integrated into larger Network Management platforms such as HP OpenView.

The menu hierarchy is similar to but not identical to the CLI on the switch. Some of the functions that EMS allows you to perform are the following:

- Manage multiple switches simultaneously
EMS has a tree control and a view panel similar to Windows Explorer. View panels can be “torn off” and docked back into the view. EMS allows you to view multiple aspects of a single switch, or compare the same view of two different switches.
- Configure the switch
EMS allows you to configure all the features on the switch, except security.
- View switch summary
EMS provides an overall summary of the switch, such as port level statistics, layer 3 statistics, server load balancing health, Syslog viewer, MP statistics, and sessions.
- View server load balancing summary
EMS allows you see the hierarchical association of server load balancing components and at the same time view their state and statistics. Color is used to give visual clues as to the health of the real server.
- Enable/disable real servers
- Configure filters
EMS allows you to insert a new filter, move a block of filters, or assign filters to ports.
- Monitor the switch
EMS monitors data in tables and graphs. You must first select a few items in the table before the graph icons are enabled. EMS generates line, bar, or pie charts.

- Export data to a file such as an Excel spreadsheet

The following screen shows a summary of the Server Load Balancing configuration based on the real servers. In this view you can see the affected services when a real server goes down. This view can also be summarized based on the state of the virtual servers.

The screenshot displays the Alteon EMS interface with the 'SLB Summary View' selected. The left pane shows a tree structure under 'Alteon EMS' with folders for 'Switch' and several IP addresses. The main pane shows a detailed view of the SLB configuration, listing objects, their states, and associated statistics.

Object	State	Sessions/s	Bytes/s
172.31.3.10 (Real Server 1)	Running	154	220,188
(Real Group 1)	Degraded	424	1,562,276
Port 80 (Service 1)	Degraded		
172.31.4.200 (Virtual Server 1)	Degraded	562	803,123
Backup Group - (Real Group 1)	Degraded	424	1,562,276
172.31.3.11 (Real Server 2)	Running	154	189,591
(Real Group 1)	Degraded	424	1,562,276
Port 80 (Service 1)	Degraded		
172.31.4.200 (Virtual Server 1)	Degraded	562	803,123
Backup Group - (Real Group 1)	Degraded	424	1,562,276
172.31.3.12 (Real Server 3)	Overflowing	100	351,805
(Real Group 1)	Degraded	424	1,562,276
Port 80 (Service 1)	Degraded		
172.31.4.200 (Virtual Server 1)	Degraded	562	803,123
Backup Group - (Real Group 1)	Degraded	424	1,562,276
172.31.3.13 (Real Server 4)	Running	108	403,774
(Real Group 1)	Degraded	424	1,562,276
Port 80 (Service 1)	Degraded		
172.31.4.200 (Virtual Server 1)	Degraded	562	803,123
Backup Group - (Real Group 1)	Degraded	424	1,562,276
197.1.1.30 (Real Server 22)	Failed	0	0
(Real Group 1)	Degraded	424	1,562,276
Port 80 (Service 1)	Degraded		
172.31.4.200 (Virtual Server 1)	Degraded	562	803,123
Backup Group - (Real Group 1)	Degraded	424	1,562,276

The bottom of the window features a toolbar with various icons and a 'Poll Interval: 2s' dropdown menu.

Using the Browser-Based Interface

Alteon OS Browser-based Interface (BBI) is a Web-based management interface for interactive network access through your Web browser.

To enable BBI access on the switch, use the command `cfg/sys/wport ena` and for more information, refer to the *BBI Quick Guide*.

Using SNMP

Alteon OS provides SNMP v1.0 support for access through any network management software, such as HP-OpenView.

To access the SNMP agent on the Alteon Application Switch, the read and write community strings on the SNMP manager should be configured to match those on the switch. The default read community string on the switch is `public` and the default write community string is `private`.

The read and write community strings on the switch can be changed using the following commands on the CLI:

```
>> /cfg/snmp/rcomm
```

and

```
>> /cfg/snmp/wcomm
```

The SNMP manager should be able to reach the management interface (management port) or any one of the IP interfaces on the switch.

For the SNMP manager to receive the traps sent out by the SNMP agent on the switch, the trap host on the switch should be configured with the following command:

```
/cfg/snmp/<trap1|trap2>
```

Alteon OS supports the following MIBs on the Alteon Application Switch:

- MIB II (RFC 1213)
- Bridge MIB (RFC 1493)
- Interface MIB (RFC 1573)

- Ethernet MIB (RFC 1643)
- BGP MIB (RFC 1657)
- OSPF MIB (RFC 1850)

For more information on using SNMP, see the *Alteon OS Command Reference*.

Using the Management Port

The management port is a Fast Ethernet port on the Alteon Application Switch that is used exclusively for managing the switch. While the switch can be managed from any network port, the management port conserves a data port that could otherwise be used for processing requests. You can use the management port to access the switch using Telnet (CLI), SSH, SNMP (EMS), or HTTP (Alteon OS Browser-Based Interface—Alteon OS BBI).

The management port does not participate in the switching and routing protocols that run on the data ports, but it can be used to perform management functions such as,

- accessing the NTP server
- sending out SNMP traps
- sending out Syslog messages
- accessing the Radius server
- accessing the DNS server
- performing TFTP functions (ptimg, gtimg, ptcfg, gtcfg, ptdmp)
- accessing the SMTP server
- running ping, Telnet and traceroute commands

NOTE – BOOTP is not supported over the management port.

For more information on using the commands to perform these functions, see the *Alteon OS Command Reference*.

Setting Up the Management Port

1. Configure a static IP address.

```
>> # /cfg/mgmt/addr 10.10.10.1      (Configure a static IP address)
>> # /cfg/mgmt/mask 255.255.255.0   (Configure a network mask)
>> # /cfg/mgmt/broad 10.10.10.255   (Configure a broadcast address)
```

2. Enable the management port.

When the management port is enabled, you can use it to access the switch via Telnet, SSH, BBI, and SNMP (EMS), provided the commands are enabled on the switch. These commands can occur simultaneously on both the management port and the data ports.

```
>> # /cfg/mgmt/ena                  (Enable the management port)
```

NOTE – You can have a maximum of four Telnet sessions on the Alteon Application Switch over the management port and the data ports together.

3. Configure the default port type for each management function.

Select the management port or the default data port for each management function. For example, select the management port for NTP, Radius, and Syslog functions only. SMTP, TFTP, and SNMP traps are configured to use the default data ports.

```
>> # /cfg/mgmt/ntp mgmt              (Select the management port for NTP)
>> # /cfg/mgmt/radius mgmt           (Select the management port for
                                     radius)
>> # /cfg/mgmt/syslog mgmt           (Select the management port for
                                     syslog)
```

NOTE – The default for TFTP can be overridden by using the `-data` or `-mgmt` option after a `gting`, `ptimg`, `gtcfg`, `ptcfg`, or `ptdmp` command.

4. Apply, verify your configuration, and save the changes.

```
>> Management Port # apply          (Make your changes active)
>> Management Port # cur             (View current settings)
>> Management Port # save            (Save for restore after reboot)
```

Securing the Switch

Secure switch management is needed for environments that perform significant management functions across the Internet. The following are some of the functions for secured management:

- Authentication of remote administrators

Authentication is the action of determining and verifying who the administrator is; it usually involves a name and a password. The password can be either a fixed password or a challenge-response query.

- Authorization of remote administrators

Once an administrator has been authenticated, authorization is the action of determining what that user is allowed to do. Authorization does not merely provide yes or no answers but may also customize the service for a particular administrator.

- Encryption of management information exchanged between the remote administrator and the switch

Examples of protocols to encrypt management information are SSH (Secure Shell) and SCP (Secure Copy). For more information, see [“Secure Shell and Secure Copy” on page 37](#).

The following sections are addressed in this section:

- [“Setting Allowable Source IP Address Ranges” on page 33](#)
- [“RADIUS Authentication and Authorization” on page 34](#)
- [“Secure Shell and Secure Copy” on page 37](#)

Setting Allowable Source IP Address Ranges

To limit access to the switch without having to configure filters for each switch port, you can set a source IP address (or range) that will be allowed to connect to the switch IP interface through Telnet, SSH, SNMP, or the Alteon OS Browser-Based Interface (BBI). This will also help to prevent spoofing or attacks on the switch's TCP/IP stack.

When an IP packet reaches the application switch, the source IP address is checked against the range of addresses defined by the management network and mask, (`mnet` and `mmask`). If the source IP address of the host or hosts are within this range, they are allowed to attempt to log in. Any packet addressed to a switch IP interface with a source IP address outside this range is discarded.

The allowable management IP address range is configured using the system `mnet` and `mmask` options available in the `/cfg/sys` menu on the Command Line Interface (CLI). Do not use the `mnet` and `mmask` commands in the `/cfg/slb/adv` menu.

For example, set `mnet` to 192.192.192.0 and `mmask` to 255.255.255.128. This defines the following range of allowed IP addresses: 192.192.192.1 to 192.192.192.127. The following source IP addresses are granted or not granted access to the switch:

- A host with a source IP address of 192.192.192.21 falls within the defined range and would be allowed to access the switch.
- A host with a source IP address of 192.192.192.192 falls outside the defined range and is not granted access. To make this source IP address valid, you would need to shift the host to an IP address within the valid range specified by the `mnet` and `mmask` or modify the `mnet` to be 192.192.192.128 and the `mmask` to be 255.255.255.128. This would put the 192.192.192.192 host within the valid range allowed by the `mnet` and `mmask` (192.192.192.128-255).

RADIUS Authentication and Authorization

Alteon OS supports the RADIUS (Remote Authentication Dial-in User Service) method to authenticate and authorize remote administrators for managing the switch. This method is based on a client/server model. The Remote Access Server (RAS)—the switch—is a client to the back-end database server. A remote user (the remote administrator) interacts only with the RAS, not the back-end server and database.

RADIUS authentication consists of the following components:

- A protocol with a frame format that utilizes UDP over IP (based on RFC 2138 and 2866)
- A centralized server that stores all the user authorization information
- A client, in this case, the switch

In [Figure 1-1](#), the Alteon Application Switch—acting as the RADIUS client—communicates to the RADIUS server to authenticate and authorize a remote administrator using the protocol definitions specified in RFC 2138 and 2866. Transactions between the client and the RADIUS server are authenticated using a shared key that is not sent over the network. In addition, the remote administrator passwords are sent encrypted between the RADIUS client (the switch) and the back-end RADIUS server.

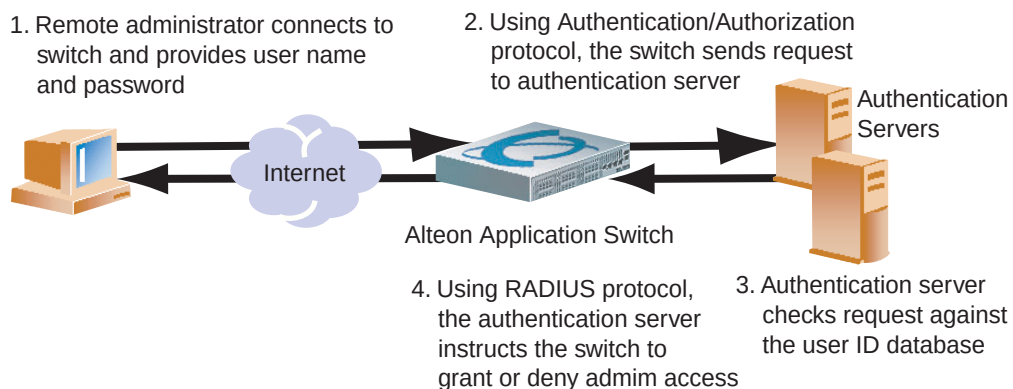


Figure 1-1 RADIUS Authentication and Authorization: How It Works

RADIUS Authentication Features in Alteon OS

Alteon OS supports the following Radius authentication features:

- Supports Radius client on the switch, based on the protocol definitions in RFC 2138 and RFC 2866.
- Allows RADIUS secret password up to 32 bytes and less than 16 octets.
- Supports *secondary authentication server* so that when the primary authentication server is unreachable, the switch can send client authentication requests to the secondary authentication server. Use the `/cfg/sys/radius/cur` command to show the currently active RADIUS authentication server.
- Supports user-configurable RADIUS server retry and time-out values:
 - Time-out value = 1-10 seconds
 - Retries = 1-3

The switch will time out if it does not receive a response from the RADIUS server in 1-3 retries. The switch will also automatically retry connecting to the RADIUS server before it declares the server down.

- Supports user-configurable RADIUS application port.
The default is 1645/UDP-based on RFC 2138. Port 1812 is also supported.
- Allows network administrator to define privileges for one or more specific users to access the switch at the RADIUS user database.
- SecurID is supported if the RADIUS server can do an ACE/Server client proxy. The password is the PIN number, plus the token code of the SecurID card.

Alteon Switch User Accounts

The user accounts listed in [Table 1-1](#) can be defined in the RADIUS server dictionary file.

Table 1-1 User Access Levels

User Account	Description and Tasks Performed	Password
User	The User has no direct responsibility for switch management. He/she can view all switch status information and statistics but cannot make any configuration changes to the switch.	user
SLB Operator	The SLB Operator manages content servers and other Internet services and their loads. In addition to being able to view all switch information and statistics, the SLB Operator can enable/disable servers using the SLB operation menu.	slboper
Layer 4 Operator	The Layer 4 Operator manages traffic on the lines leading to the shared Internet services. This user currently has the same access level as the SLB operator. This level is reserved for future use, to provide access to operational commands for operators managing traffic on the line leading to the shared Internet services.	l4oper
Operator	The Operator manages all functions of the switch. In addition to SLB Operator functions, the Operator can reset ports or the entire switch.	oper
SLB Administrator	The SLB Administrator configures and manages content servers and other Internet services and their loads. In addition to SLB Operator functions, the SLB Administrator can configure parameters on the SLB menus, with the exception of not being able to configure filters or bandwidth management.	slbadmin
Layer 4 Administrator	The Layer 4 Administrator configures and manages traffic on the lines leading to the shared Internet services. In addition to SLB Administrator functions, the Layer 4 Administrator can configure all parameters on the SLB menus, including filters and bandwidth management.	l4admin
Administrator	The super-user Administrator has complete access to all menus, information, and configuration commands on the switch, including the ability to change both the user and administrator passwords.	admin

RADIUS Attributes for Alteon OS User Privileges

When the user logs in, the switch authenticates his/her level of access by sending the RADIUS access request, that is, the client authentication request, to the RADIUS authentication server.

If the remote user is successfully authenticated by the authentication server, the switch will verify the *privileges* of the remote user and authorize the appropriate access. When both the primary and secondary authentication servers are not reachable, the administrator has an option to allow *backdoor* access via the console only or console and Telnet access. The default is *disable* for Telnet access and *enable* for console access.

All user privileges, other than those assigned to the Administrator, have to be defined in the RADIUS dictionary. Radius attribute 6 which is built into all Radius servers defines the administrator. The file name of the dictionary is RADIUS vendor-dependent. The following Radius attributes are defined for Alteon OS user privileges levels:

Table 1-2 Alteon OS-proprietary Attributes for Radius

User Name/Access	User-Service-Type	Value
User	<i>Vendor-supplied</i>	255
SLB Operator	<i>Vendor-supplied</i>	254
Layer 4 Operator	<i>Vendor-supplied</i>	253
Operator	<i>Vendor-supplied</i>	252
SLB Administrator	<i>Vendor-supplied</i>	251
Layer 4 Administrator	<i>Vendor-supplied</i>	250

Secure Shell and Secure Copy

The Telnet method of managing an Alteon Application Switch does not provide a secure connection. Secure Shell (SSH) and Secure Copy (SCP) however, use secure tunnels so that messages between a remote administrator and the switch is encrypted and secured.

SSH is a protocol that enables remote administrators to log securely into another computer over a network to execute management commands. **SCP** is typically used to copy files securely from one machine to another. SCP uses SSH for encryption of data on the network. On an Alteon Application Switch, SCP is used to download and upload the switch configuration via secure channels.

The benefits of using SSH and SCP are listed below:

- Authentication of remote administrators
Identifying the administrator using Name/Password
- Authorization of remote administrators
Determining the permitted actions and customizing service for individual administrators
- Encryption of management messages
Encrypting messages between the remote administrator and switch
- Secure copy support

The Alteon OS implementation of SSH is based on version 1.5 and supports SSH clients version 1.5—1.x.xx only. The following SSH clients have been tested:

- SSH 1.2.23 and SSH 1.2.27 for Linux (freeware)
- SecureCRT 3.0.2 and SecureCRT 3.0.3 for Windows NT (Van Dyke Technologies, Inc.)
- F-Secure SSH 1.1 for Windows (Data Fellows)

NOTE – There can be a maximum number of four simultaneous Telnet/SSH/SCP connections at one time. The `/cfg/sys/radius/telnet` command also applies to SSH/SCP connections.

Encryption of Management Messages

The following encryption and authentication methods are supported for SSH and SCP:

Server Host Authentication:	Client RSA authenticates the switch at the beginning of every connection
Key Exchange:	RSA
Encryption:	3DES-CBC, DES
User Authentication:	Local password authentication, RADIUS, SecurID (via RADIUS, for SSH only—does not apply to SCP)

SCP Services

To perform SCP commands, you need the SCP admin password with administrator privileges (this password must be different from the admin password).

The following SCP commands are supported in this service. These commands are entered using the CLI on the client that is running the SCP application:

- `getcfg` is used to download the switch's configuration to the remote host via SCP.
- `putcfg` is used to upload the switch's configuration from a remote host to the switch; the `diff` command is automatically executed at the end of `putcfg` to notify the remote client of the difference between the new and the current configurations.
- `putcfg_apply` runs the `apply` command after the `putcfg` is done.
- `putcfg_apply_save` saves the new configuration to the flash after `putcfg_apply` is done.

The `putcfg_apply` and `putcfg_apply_save` commands are provided because extra `apply` and `save` commands are usually required after a `putcfg`; however, an SCP session is not in an interactive mode at all.

RSA Host and Server Keys

To support the SSH server feature, two sets of RSA keys (host and server keys) are required. The host key is 1024 bits and is used to identify the Alteon Application switch. The server key is 768 bits and is used to make it impossible to decipher a captured session by breaking into the Alteon switch at a later time.

When the SSH server is first enabled and applied, the switch automatically generates the RSA host and server keys and is stored in the FLASH memory. However, you may use the following commands to generate them manually:

<code>>> # /cfg/sys/sshd/hkeygen</code>	<i>(Generates the host key)</i>
<code>>> # /cfg/sys/sshd/skeygen</code>	<i>(Generates the server key)</i>

These two commands take effect immediately without the need of an apply command.

When the switch reboots, it will retrieve the host and server keys from the FLASH memory. If these two keys are not available in the flash and if the SSH server feature is enabled, the switch automatically generates them during the system reboot. **This process may take several minutes to complete.**

The switch can also automatically regenerate the RSA server key. To set the interval of RSA server key autogeneration, use this command:

<code>>> # /cfg/sys/sshd/interval <number of hours (0-24)></code>

where the number of hours must range between 0–24, and a value of 0 denotes that RSA server key autogeneration is disabled. When greater than 0, the switch will autogenerate the RSA server key every specified interval; however, RSA server key generation is skipped if the switch is busy doing other key or cipher generation when the timer expires.

NOTE – The Alteon switch will perform only one session of key/cipher generation at a time. Thus, an SSH/SCP client will not be able to log in if the switch is performing key generation at that time, or if another client has logged in immediately prior. Also, key generation will fail if an SSH/SCP client is logging in at that time.

Radius Authentication

SSH/SCP is integrated with RADIUS authentication. After the RADIUS server is enabled on the switch, all subsequent SSH authentication requests will be redirected to the specified RADIUS servers for authentication. The redirection is transparent to the SSH clients.

SecurID Support

SSH/SCP can also work with SecurID, a token card-based authentication method. The use of SecurID requires the interactive mode during login, which is not provided by the SSH connection.

NOTE – There is no SNMP or Browser-Based Interface (BBI) support for SecurID because the SecurID server, ACE, is a one-time password authentication and requires an interactive session.

To log in using SSH, you need to use a special username, “ace,” to bypass the SSH authentication. After an SSH connection is established, you are prompted to enter the username and password (the SecurID authentication is being performed now). You need to provide your username and the token in your SecurID card as a regular Telnet user.

To use SCP, you need to use the SCP-only administrator’s password (that is, the `scpadm` option under the `/cfg/sys/sshd` menu) to bypass the checking of SecurID. A SCP-only administrator’s password is typically used when SecurID is used. For example, it can be used in an automation program (in which the tokens of SecurID are not available) to back up (download) the switch configurations each day.

NOTE – The SCP-only administrator’s password must be different from the regular administrator’s password. If the two passwords are the same, the administrator using that password will not be allowed to log in as a SSH user because the switch will recognize him as the SCP-only administrator and only allow the administrator access to SCP commands.

Alternately, you can configure a regular administrator with a fixed password in the RADIUS server if it can be supported. A regular administrator with a fixed password in the RADIUS server can perform both SSH and SCP with no additional authentication required.

Configuring SSH/SCP

SSH/SCP parameters can be configured via the console port only, using the CLI. However, SCP `putcfg` and TFTP `getcfg` can also change the SSH/SCP configuration. When SSH is enabled, SCP is also enabled. The switch SSH daemon uses TCP port 22 only and is not configurable.

To enable or disable the SSH/SCP feature, use the following commands:

<code>>> # /cfg/sys/sshd/on</code>	<i>(Turn SSH/SCP on)</i>
<code>>> # /cfg/sys/sshd/off</code>	<i>(Turn SSH/SCP off)</i>

To enable or disable the SCP apply and save (SCP `putcfg_apply` and `putcfg_apply_save` commands), use these commands:

```
>> # /cfg/sys/sshd/ena           (Enable SSH/SCP apply and save)
>> # /cfg/sys/sshd/dis          (Disable SSH/SCP apply and save)
```

Examples of Client Commands

The following list shows the format for using some client commands:

- To log in to the switch:
ssh <switch IP address> or **ssh -l** <username> <switch IP address>
- To download the switch configuration using SCP:
scp <switch IP address>:**getcfg** <local filename>
- To upload the configuration to the switch:
scp <local filename> <switch IP address>:**putcfg**

For example, if 205.178.15.157 is the IP address of a sample switch, then the following client commands are issued:

```
>> # ssh 205.178.15.157
>> # ssh -l <login-name> 205.178.15.157
>> # scp 205.178.15.157:getcfg ad4.cfg
>> # scp ad4.cfg 205.178.15.157:putcfg
```

The `apply` and `save` commands are still needed after the last command (`scp ad4.cfg 205.178.15.157:putcfg`). Or, instead, you can use the following commands:

```
>> # scp ad4.cfg 205.178.15.157:putcfg_apply
>> # scp ad4.cfg 205.178.15.157:putcfg_apply_save
```



CHAPTER 2

VLANs

This chapter describes network design and topology considerations for using Virtual Local Area Networks (VLANs). VLANs are commonly used to split up groups of network users into manageable broadcast domains, to create logical segmentation of workgroups, and to enforce security policies among logical segments. The following topics are discussed in this chapter:

- “VLAN ID Numbers” on page 44
- “VLAN Tagging” on page 44
- “VLANs and the IP Interfaces” on page 45

This section briefly describes how management functions can only be accomplished from stations on VLANs that include an IP interface to the switch.

- “VLAN Topologies and Design Issues” on page 45

This section discusses how you can logically connect users and segments to a host that supports many logical segments or subnets by using the flexibility of the multiple VLAN system.

- “VLANs and Default Gateways” on page 49

NOTE – Basic VLANs can be configured during initial switch configuration (see “Using the Setup Utility” in the *Alteon OS Command Reference*). More comprehensive VLAN configuration can be done from the Command Line Interface (see “VLAN Configuration” as well as “Port Configuration” in the *Alteon OS Command Reference*).

VLAN ID Numbers

Alteon OS supports up to 255 VLANs per switch. Even though the maximum number of VLANs supported at any given time is 255, each can be identified with any number between 1 and 4090.

VLANs are defined on a per-port basis. Each port on the switch can belong to one or more VLANs, and each VLAN can have any number of switch ports in its membership. Any port that belongs to multiple VLANs, however, must have *VLAN tagging* enabled (see [“VLAN Tagging” on page 44](#)).

Each port in the switch has a configurable default VLAN number, known as its *PVID*. The factory default value for all PVIDs is 1. This places all ports on the same VLAN initially, although each port's PVID is configurable to any VLAN number between 1 and 4090.

Any untagged frames (those with no VLAN specified) are classified with the sending port's PVID.

VLAN Tagging

Alteon OS software supports 802.1Q *VLAN tagging*, providing standards-based VLAN support for Ethernet systems.

Tagging places the VLAN identifier in the frame header, allowing multiple VLANs per port. When you configure multiple VLANs on a port, you must also enable tagging on that port.

Since tagging fundamentally changes the format of frames transmitted on a tagged port, you must carefully plan network designs to prevent tagged frames from being transmitted to devices that do not support 802.1Q VLAN tags.

VLANs and the IP Interfaces

Carefully consider how you create VLANs within the switch, so that communication with the switch remains possible.

You can access the switch for remote configuration, trap messages, and other management functions only from stations on VLANs that include an IP interface to the switch (see “IP Interface Menu” section in the *Alteon OS Command Reference*). Likewise, you can cut off access to management functions to any VLAN by excluding IP interfaces from the VLAN’s membership.

For example, if all IP interfaces are left on VLAN 1 (the default), and all ports are configured for VLANs other than VLAN 1, then switch management features are effectively cut off. If an IP interface is added to one of the other VLANs, the stations in that VLAN will all have access to switch management features.

VLAN Topologies and Design Issues

By default, the Alteon OS software has a single VLAN configured on every port. This configuration groups all ports into the same broadcast domain. The VLAN has an 802.1Q VLAN PVID of 1. VLAN tagging is turned off, because by default only a single VLAN is configured per port.

Since VLANs are most commonly used to create individual broadcast domains and/or separate IP subnets, host systems should be present on more than one VLAN simultaneously. Alteon Application Switches and VLAN-tagging server adapters support multiple VLANs on a per-port or per-interface basis, allowing very flexible configurations.

You can configure multiple VLANs on a single VLAN-tagging server adapter, with each VLAN being configured through a logical interface and logical IP address on the host system. Each VLAN configured on the server adapter must also be configured on the switch port to which it is connected. If multiple VLANs are configured on the port, tagging must be turned on.

Using this flexible multiple VLAN system, you can logically connect users and segments to a host with a single VLAN-tagging adapter that supports many logical segments or subnets.

NOTE – If a 802.1Q tagged frame is sent to a port that has vlan-tagging disabled, then the frames are dropped at the ingress port.

Example 1: Multiple VLANs with Tagging Adapters

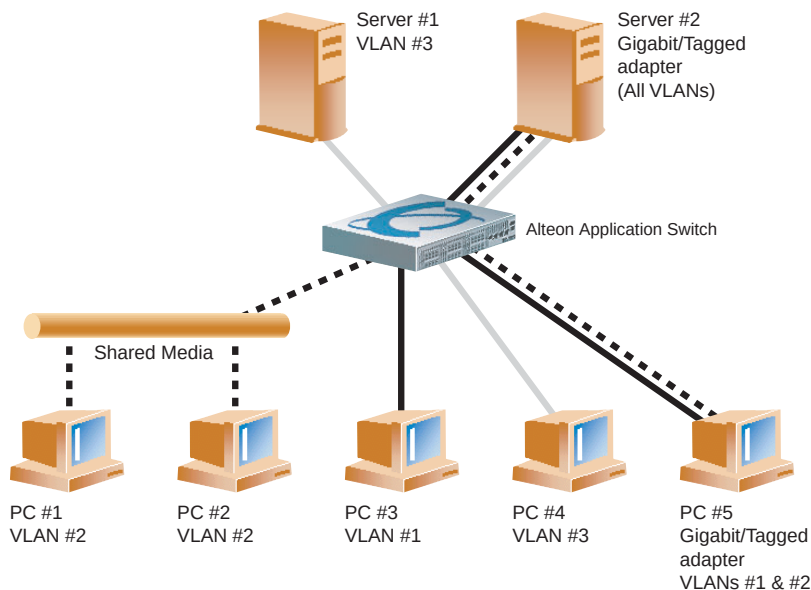


Figure 2-1 Example 1: Multiple VLANs with VLAN-Tagged Gigabit Adapters

The features of this VLAN are described below:

Component	Description
Application Switch	This switch is configured for three VLANs that represent three different IP subnets. Two servers and five clients are attached to the switch.
Server #1	This server is part of VLAN 3 and only has presence in one IP subnet. The port that the VLAN is attached to is configured only for VLAN 3, so VLAN tagging is off.
Server #2	This high-use server needs to be accessed from all VLANs and IP subnets. The server has a VLAN-tagging adapter installed with VLAN tagging turned on. The adapter is attached to one of the application switch's Gigabit Ethernet ports, that is configured for VLANs 1, 2, and 3. Tagging is turned on. Because of the VLAN tagging capabilities of both the adapter and the switch, the server is able to communicate on all three IP subnets in this network. Broadcast separation between all three VLANs and subnets, however, is maintained.

Component	Description
PCs #1 and #2	These PCs are attached to a shared media hub that is then connected to the switch. They belong to VLAN 2 and are logically in the same IP subnet as Server 2 and PC 5. Tagging is not enabled on their switch port.
PC #3	A member of VLAN 1, this PC can minimize its broadcast domain to Server 2 and PC 5.
PC #4	A member of VLAN 3, this PC can minimize its broadcast domain to Server 1 and Server 2.
PC #5	A member of both VLAN 1 and VLAN 2, this PC has VLAN-tagging Gigabit Ethernet adapter installed. It can minimize its broadcast domain to Server #2 via VLAN 1, and to PC #1 and PC #2 via VLAN 2. The switch port to which it is connected is configured for both VLAN 1 and VLAN 2 and has tagging enabled.

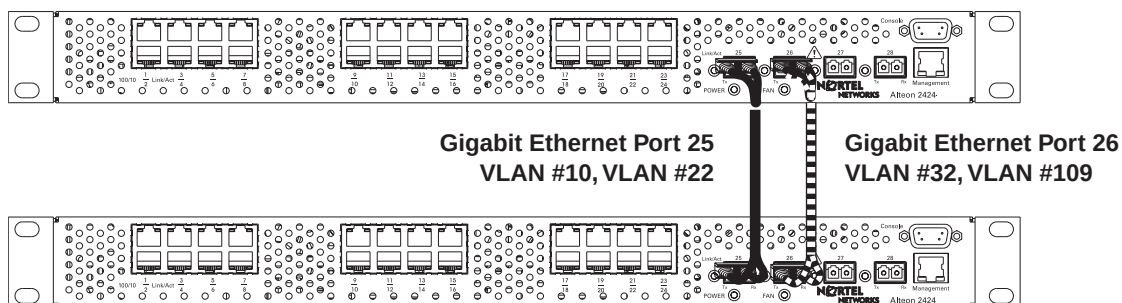
NOTE – VLAN tagging is required only on ports that are connected to other Alteon Application Switches or on ports that connect to tag-capable end-stations, such as servers with VLAN-tagging adapters.

Example 2: Parallel Links with VLANs

Example 2 shows how it is possible, through the use of VLANs, to create configurations where there are multiple links between two switches, without creating broadcast loops.

In [Figure 2-2](#), two Alteon Application switches are connected with two different Gigabit Ethernet links. Without VLANs, this configuration would create a broadcast loop. To prevent broadcast loops, port 25 is on VLAN 10, port 26 is on VLAN 109. Both switch-to-switch links are on different VLANs and, thus, are separated into their own broadcast domains.

Alteon Application Switch #1



Alteon Application Switch #2

Figure 2-2 Example 2: Parallel Links with VLANs

In this example the Gig ports are on different VLANs and Spanning Tree Protocol is disabled. For information on Spanning Tree Protocol, see [Chapter 4, “Spanning Tree Protocol.”](#)

VLANs and Default Gateways

Alteon OS allows you to assign different gateways for each VLAN. You can effectively map multiple customers to specific gateways on a single switch. The benefits of segregating customers to different default gateways are:

- Resource optimization
- Enhanced customer segmentation
- Improved service differentiation

Segregating VLAN Traffic

Deploy this feature in an environment where you want to segregate VLAN traffic to a configured default gateway. In [Figure 2-3](#), VLANs 2 and 3 have different routing requirements. VLAN 2 is required to route traffic through default gateway 5 and VLAN 3 is required to route traffic through default gateway 6.

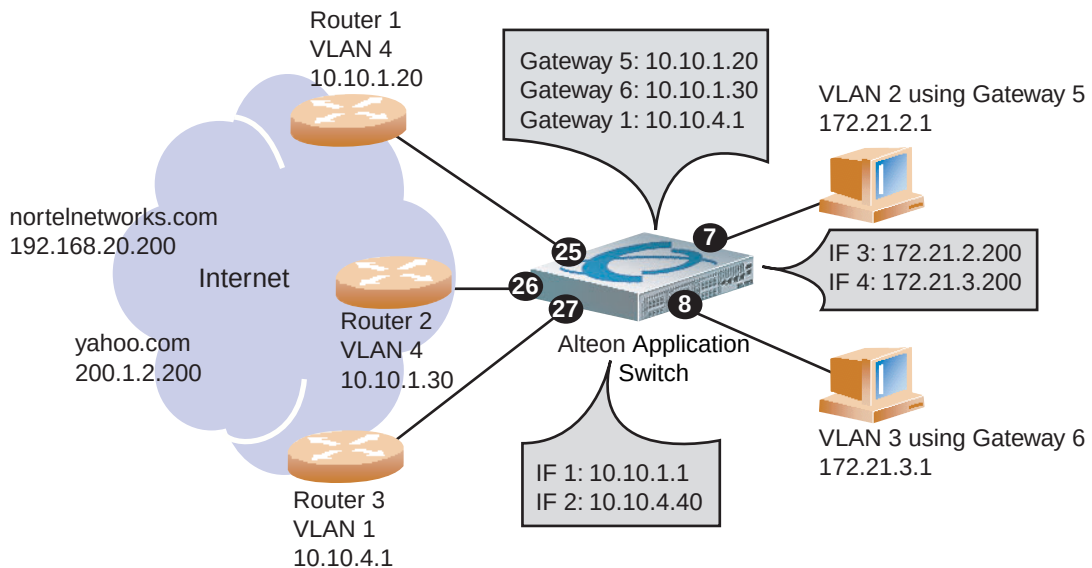


Figure 2-3 Default Gateways per VLAN

You can configure up to 255 gateways with one gateway per VLAN with values starting from 5 through 259. If the gateways per VLAN fail, then traffic is directed to default gateways 1 through 4. Default gateways 1 through 4 are used for load balancing session requests and as backup when a specific gateway that has been assigned to a VLAN is down.

In the example shown in [Figure 2-3](#), if gateways 5 or 6 fail, then traffic is directed to default gateway 1, which is configured with IP address 10.10.4.1. If default gateways 1 through 4 are not configured on the switch, then packets from VLAN 2 and VLAN 3 are discarded.

The route cache table on the switch records each session request by mapping the destination IP address with the MAC address of the default gateway. The command `/info/arp/dump` on the switch command line will display the entries in the route cache similar to those shown in [Table 2-1](#). The destination IP addresses (see the last two rows) are associated with the MAC addresses of the gateways.

Table 2-1 Route Cache Example

Destination IP address	Flags	MAC address	VLAN	Port	Referenced SPs
10.10.1.1	P	00:60:cf:46:48:60	4		1-4
10.10.1.20		00:60:cf:44:cd:a0	4	25 (Gig)	empty
10.10.1.30		00:60:cf:42:3b:40	4	26 (Gig)	empty
10.10.4.1		00:60:cf:42:77:e0	1	27 (Gig)	empty
10.10.4.40	P	00:60:cf:46:48:60	1		1-4
172.21.2.27		00:50:da:17:c8:05	2	7	1
172.21.2.200	P	00:60:cf:46:48:60	2		1-4
172.21.3.14		00:c0:4f:09:3e:56	3	8	2
172.21.2.200	P	00:60:cf:46:48:60	3		1-4
192.168.20.200	R	00:60:cf:44:cd:a0	4	1	7
200.1.2.200	R	00:60:cf:42:3b:40	4	2	8

As shown in [Table 2-1](#), traffic from VLAN 2 uses Gateway 5 to access destination IP address 192.168.20.200. If traffic from VLAN 3 requests the same destination address, then traffic is routed via Gateway 5 instead of Gateway 6, because 192.168.20.200 in the route cache is mapped to Gateway 5. If the requested route is not in the route cache, then the switch reads the routing table. If the requested route is not in the routing table, then the switch looks at the configured default Gateway.

Configuring the Local Network

To completely segregate VLAN traffic to its own default gateway, you can configure the local network addresses of the VLAN. This will ensure that all traffic from VLAN 2 is forwarded to Gateway 5 and all traffic from VLAN 3 is forwarded to Gateway 6.

Typically, the switch routes traffic based on the routes in the routing table. The routing table will contain an entry of the configured local network with the default gateway. The route cache will not contain the route entry. This configuration provides a more secure environment, but affects performance if the routing table is close to its maximum capacity.

Configuring Gateways per VLAN

Follow this procedure to configure the example shown in [Figure 2-3 on page 49](#):

1. Assign an IP address for each router and client workstation.
2. Assign an IP interface for each subnet attached to the switch.

>> /cfg/ip/IF 1	(Select IP interface 1 for gateway 5 & 6 subnet)
>> IP Interface 1# addr 10.10.1.1	(Assign IP address for interface 1)
>> IP Interface 1# mask 255.255.255.0	(Assign mask for IF 1)
>> IP Interface 1# broad 10.10.1.255	(Assign broadcast address for IF 1)
>> IP Interface 1# vlan 4	(Assign VLAN 4 to IF 1)
>> IP Interface 1# ../if 2	(Select IP interface 2 for gateway 1)
>> IP Interface 2# addr 10.10.4.40	(Assign IP address for interface 2)
>> IP Interface 2# mask 255.255.255.0	(Assign mask for IF 2)
>> IP Interface 2# broad 10.10.4.255	(Assign broadcast address for IF 2)
>> IP Interface 2# vlan 1	(Assign VLAN 1 to IF 2)
>> IP Interface 2# ../if 3	(Select IP interface 3 for VLAN 2 subnet)
>> IP Interface 3# addr 172.21.2.200	(Assign IP address for interface 3)
>> IP Interface 3# mask 255.255.255.0	(Assign mask for IF 3)
>> IP Interface 3# broad 172.21.2.255	(Assign broadcast address for IF 3)
>> IP Interface 3# vlan 2	(Assign VLAN 2 to IF 3)
>> IP Interface 3# ../if 4	(Select IP interface 4 for VLAN 3 subnet)
>> IP Interface 4# addr 172.21.3.200	(Assign IP address for interface 4)
>> IP Interface 4# mask 255.255.255.0	(Assign mask for IF 4)
>> IP Interface 4# broad 172.21.3.255	(Assign broadcast address for IF 4)
>> IP Interface 4# vlan 3	(Assign VLAN 3 to IF 4)

3. Configure the default gateways.

Configuring gateways 5 and 6 for VLANs 2 and 3 respectively. Configure default gateway 1 for load balancing session requests and as backup when gateways 5 and 6 fail.

>> /cfg/ip/gw 5	(Select gateway 5)
>> Default gateway 5# addr 10.10.1.20	(Assign IP address for gateway 5)
>> Default gateway 5# ../gw 6	(Select default gateway 6)
>> Default gateway 6# addr 10.10.1.30	(Assign IP address for gateway 6)
>> Default gateway 6# ../gw 1	(Select default gateway 1)
>> Default gateway 1# addr 10.10.4.1	(Assign IP address for gateway 1)

NOTE – The IP address for default gateways 1 to 4 must be unique. IP addresses for default gateways 5 to 259 can be set to the same IP address as the other gateways (including default gateway 1 to 4). For example, you can configure two default gateways with the same IP address for two different VLANs.

4. Add the VLANs to the gateways and enable them.

>> /cfg/ip/gw 5	(Select gateway 5)
>> Default gateway 5# vlan 2	(Add VLAN 2 for default gateway 5)
>> Default gateway 5# ena	(Enable gateway 5)
>> Default gateway 5# ../ gw 6	(Select gateway 6)
>> Default gateway 6# vlan 3	(Add VLAN 3 for default gateway 6)
>> Default gateway 6# ena	(Enable gateway 6)
>> Default gateway 6# ../gw 1	(Select default gateway 1)
>> Default gateway 1# ena	(Enable gateway 1 for all VLAN s)

5. Apply and verify your configuration.

>> Default gateway 1# ../cur	(View current IP settings)

Examine the results under the gateway section. If any settings are incorrect, make the appropriate changes.

6. (Optional) Configure the local networks to ensure that the VLANs use the configured default gateways.

>> IP# frwd/local	<i>(Select the local network Menu)</i>
>> IP Forwarding# add 10.10.0.0	<i>(Specify the network for routers 1, 2, & 3)</i>
>> IP Forwarding# mask 255.255.0.0	<i>(Add the mask for the routers)</i>
>> IP Forwarding# add 172.21.2.0	<i>(Specify the network for VLAN 2)</i>
>> IP Forwarding# mask 255.255.255.0	<i>(Add the mask for VLAN 2 network)</i>
>> IP Forwarding# add 172.21.3.0	<i>(Specify the network for VLAN 3)</i>
>> IP Forwarding# mask 255.255.255.0	<i>(Add the mask for VLAN 3)</i>

7. Apply and save your new configuration changes.

>> IP Forwarding# apply
>> IP Forwarding# save

CHAPTER 3

Port Trunking

Trunk groups can provide super-bandwidth, multi-link connections between Alteon Application Switches or other trunk-capable devices. A trunk group is a group of ports that act together, combining their bandwidth to create a single, larger virtual link. This chapter provides configuration background and examples for trunking multiple ports together:

- “[Overview](#)” on this page
- “[Port Trunking Example](#)” on page 57

Overview

When using port trunk groups between two Alteon Application Switches as shown in [Figure 3-1](#), you can create a virtual link between the switches operating up to 4 Gig per second, depending on how many physical ports are combined. The switch supports up to 12 trunk groups per switch, each with two to eight ports per group.

Alteon Application Switch

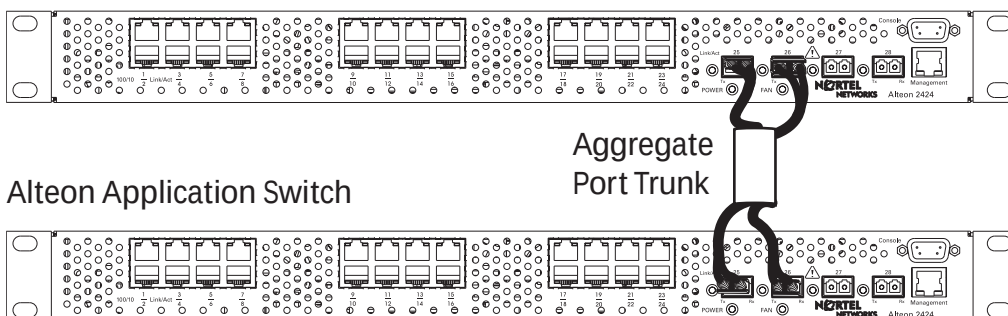


Figure 3-1 Port Trunk Group

Trunk groups are also useful for connecting an Alteon Application Switch to third-party devices that support link aggregation, such as Cisco routers and switches with EtherChannel technology (*not* ISL trunking technology) and Sun's Quad Fast Ethernet Adapter. Nortel Networks trunk group technology is compatible with these devices when they are configured manually.

Statistical Load Distribution

Network traffic is statistically load balanced between the ports in a trunk group. The Alteon OS-powered switch uses both the Layer 2 MAC address and Layer 3 IP address information present in each transmitted frame for determining load distribution.

The addition of Layer 3 IP address examination is an important advance for traffic distribution in trunk groups. In some port trunking systems, only Layer 2 MAC addresses are considered in the distribution algorithm. Each packet's particular combination of source and destination MAC addresses results in selecting one line in the trunk group for data transmission. If there are enough Layer 2 devices feeding the trunk lines, then traffic distribution becomes relatively even. In some topologies, however, only a limited number of Layer 2 devices (such as a handful of routers and servers) feed the trunk lines. When this occurs, the limited number of MAC address combinations encountered results in a lopsided traffic distribution, which can reduce the effective combined bandwidth of the trunked ports.

By adding Layer 3 IP address information to the distribution algorithm, a far wider variety of address combinations is seen. Even with just a few routers feeding the trunk, the normal source/destination IP address combinations (even within a single LAN) can be widely varied. This results in a wider statistical load distribution and maximizes the use of the combined bandwidth available to trunked ports.

Built-In Fault Tolerance

Since each trunk group is comprised of multiple physical links, the trunk group is inherently fault tolerant. As long as one connection between the switches is available, the trunk remains active.

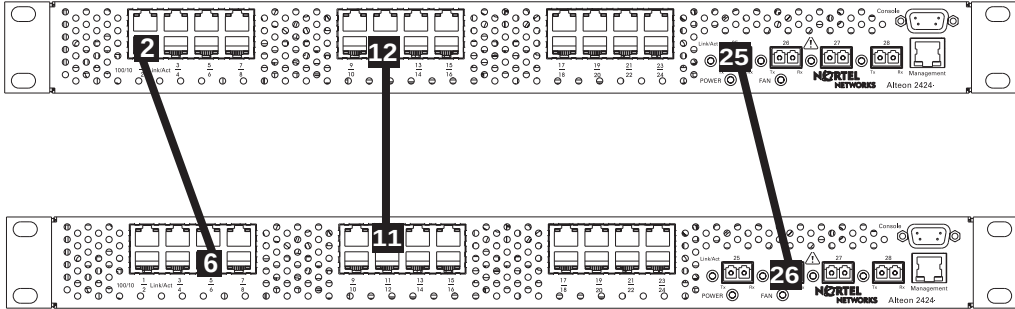
Statistical load balancing is maintained whenever a port in a trunk group is lost or returned to service.

Port Trunking Example

In the example below, three ports will be trunked between two Alteon Application Switches.

Alteon Application Switch #1

Trunk 1: Ports 2, 12, and 25 on Switch 1



Alteon Application Switch #2

Trunk 3: Ports 6, 11, and 26 on Switch 2

Figure 3-2 Port Trunk Group Configuration Example

Prior to configuring each switch in the above example, you must connect to the appropriate switch's Command Line Interface (CLI) as the administrator.

NOTE – For details about accessing and using any of the menu commands described in this example, see the *Alteon OS Command Reference*.

1. **Connect the switch ports that will be involved in the trunk group.**
2. **Follow these steps on application switch 1:**
 - (a) Define a trunk group.

>> # /cfg/trunk 1	(Select trunk group 1)
>> Trunk group 1# add 2	(Add port 2 to trunk group 1)
>> Trunk group 1# add 12	(Add port 12 to trunk group 1)
>> Trunk group 1# add 25	(Add port 25 to trunk group 1)
>> Trunk group 1# ena	(Enable trunk group 1)

(b) Apply and verify the configuration.

>> Trunk group 1# apply	<i>(Make your changes active)</i>
>> Trunk group 1# cur	<i>(View current trunking configuration)</i>

Examine the resulting information. If any settings are incorrect, make appropriate changes.

(c) Save your new configuration changes.

>> Trunk group 1# save	<i>(Save for restore after reboot)</i>

3. Repeat the process on application switch 2.

>> # /cfg/trunk 3	<i>(Select trunk group 3)</i>
>> Trunk group 3# add 6	<i>(Add port 6 to trunk group 3)</i>
>> Trunk group 3# add 11	<i>(Add port 11 to trunk group 3)</i>
>> Trunk group 3# add 26	<i>(Add port 26 to trunk group 3)</i>
>> Trunk group 3# ena	<i>(Enable trunk group 3)</i>
>> Trunk group 3# apply	<i>(Make your changes active)</i>
>> Trunk group 3# cur	<i>(View current trunking configuration)</i>
>> Trunk group 3# save	<i>(Save for restore after reboot)</i>

Trunk group 1 (on application switch 1) is now connected to trunk group 3 (on application switch 2).

NOTE – In this example, two Alteon Application Switches are used. If a third-party device supporting link aggregation is used (such as Cisco routers and switches with EtherChannel technology or Sun's Quad Fast Ethernet Adapter), trunk groups on the third-party device should be configured manually. Connection problems could arise when using automatic trunk group negotiation on the third-party device.

4. Examine the trunking information on each switch.

```
>> /info/trunk
```

(View trunking information)

Information about each port in each configured trunk group will be displayed. Make sure that trunk groups consist of the expected ports and that each port is in the expected state.

The following restrictions apply:

- Any physical switch port can belong to only one trunk group.
- Up to eight ports can belong to the same trunk group.
- Best performance is achieved when all ports in any given trunk group are configured for the same speed.
- Trunking from non-Alteon devices must comply with Cisco® EtherChannel® technology.



CHAPTER 4

Spanning Tree Protocol

When multiple paths exist on a network, Spanning Tree Protocol (STP) configures the network so that a switch uses only the most efficient path. The following topics are discussed in this chapter:

- “Overview” on page 62
- “Bridge Protocol Data Units (BPDUs)” on page 63
- “Multiple Spanning Trees” on page 64

Overview

Spanning Tree Protocol (STP) detects and eliminates logical loops in a bridged or switched network. STP forces redundant data paths into a standby (blocked) state. When multiple paths exist, Spanning Tree configures the network so that a switch uses only the most efficient path. If that path fails, Spanning Tree automatically sets up another active path on the network to sustain network operations.

The relationship between port, trunk groups, VLANs, and Spanning Trees is shown in [Table 4-1](#).

Table 4-1 Ports, Trunk Groups, and VLANs

Switch Element	Belongs to
Port	Trunk group or One or more VLANs
Trunk group	One or more VLANs
VLAN	One Spanning Tree group

NOTE – Due to Spanning Tree’s sequence of listening, learning, and forwarding or blocking, lengthy delays may occur. For more information on using STP in cross-redundant topologies, see [“Eliminating Loops with STP and VLANs”](#) on page 317.

Bridge Protocol Data Units (BPDUs)

To create a Spanning Tree, the application switch generates a configuration Bridge Protocol Data Unit (BPDU), which it then forwards out of its ports. All switches in the Layer 2 network participating in the Spanning Tree gather information about other switches in the network through an exchange of BPDUs.

A BPDU is a 64-byte packet that is sent out at a configurable interval, which is typically set for two seconds. The BPDU is used to establish a path, much like a “hello” packet in IP routing. BPDUs contain information about the transmitting bridge and its ports, including bridge and MAC addresses, bridge priority, port priority, and path cost. If the ports are tagged, each port sends out a special BPDU containing the tagged information.

The generic action of a switch on receiving a BPDU is to compare the received BPDU to its own BPDU that it will transmit. If the received BPDU is better than its own BPDU, it will replace its BPDU with the received BPDU. Then, the application switch adds its own bridge ID number and increments the path cost of the BPDU. The application switch uses this information to block any necessary ports.

Determining the Path for Forwarding BPDUs

When determining which port to use for forwarding and which port to block, application switches use information in the BPDU, including each bridge priority ID. A technique based on the “lowest root cost” is then computed to determine the most efficient path for forwarding.

For more information on bridge priority, port priority, and port cost, refer to the *Alteon OS Command Reference*. Much like least-cost routing, root cost assigns lower values to high-bandwidth ports, such as Gigabit Ethernet, to encourage their use. For example, a 10-Mbps link has a “cost” of 100, a 100-Mbps (Fast Ethernet) link carries a cost of 10, and a 1000-Mbps (or Gigabit Ethernet) link has a cost of 1. The objective is to use the fastest links so that the route with the lowest cost is chosen.

Multiple Spanning Trees

Alteon OS supports up to 16 instances of Spanning Trees or Spanning Tree groups. Each VLAN can be placed in only one Spanning Tree group per switch except for the default Spanning Tree group (STG 1). The default Spanning Tree group (1) can have more than one VLAN. All other Spanning Tree groups (2-16) can have only one VLAN associated with it. Spanning Tree can be enabled or disabled for each port. Multiple Spanning Trees can be enabled on tagged or untagged ports.

NOTE – By default, all newly created VLANs are members of Spanning Tree Group 1.

Why Do We Need Multiple Spanning Trees?

Figure 4-1 shows a simple example of why we need multiple Spanning Trees. Two VLANs, VLAN 1 and VLAN 100 exist between Alteon Application Switch A and Alteon Application Switch B. If you have a single Spanning Tree group, the switches see an apparent loop, and one VLAN may become blocked, affecting connectivity, even though no actual loop exists.

If VLAN 1 and VLAN 100 belong to different Spanning Tree Groups, then the two instances of Spanning Tree separate the topology without forming a loop. Both VLANs can forward packets between the application switches without losing connectivity.

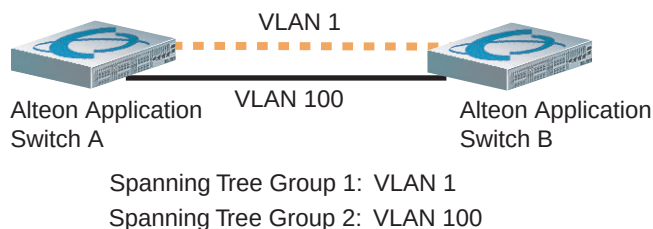


Figure 4-1 Using Multiple Instances of Spanning Tree Protocol

Four-Switch Topology with a Single Spanning Tree

In the four-switch topology example shown in Figure 4-2 on page 65, and assuming Alteon switch A has a higher priority, you can have at least three loops on the network:

- Data flowing from application switches A to B to C and back to application switch A.
- Data flowing from application switches A to C to D and back to application switch A

- Data flowing from application switches A to B to C to D and back to application switch A.

With a single Spanning Tree environment, as shown in [Figure 4-2](#), you will have two links blocked to prevent loops on the network. It is possible that the blocks may be between application switches C and D and between application switches B and C, depending on the bridge priority, port priority, and port cost. The two blocks would prevent looping on the network, but the blocked link between application switches B and C will inadvertently isolate VLAN 3 altogether.

NOTE – For more information on bridge priority, port priority, and port cost see the *Alteon OS Command Reference*.

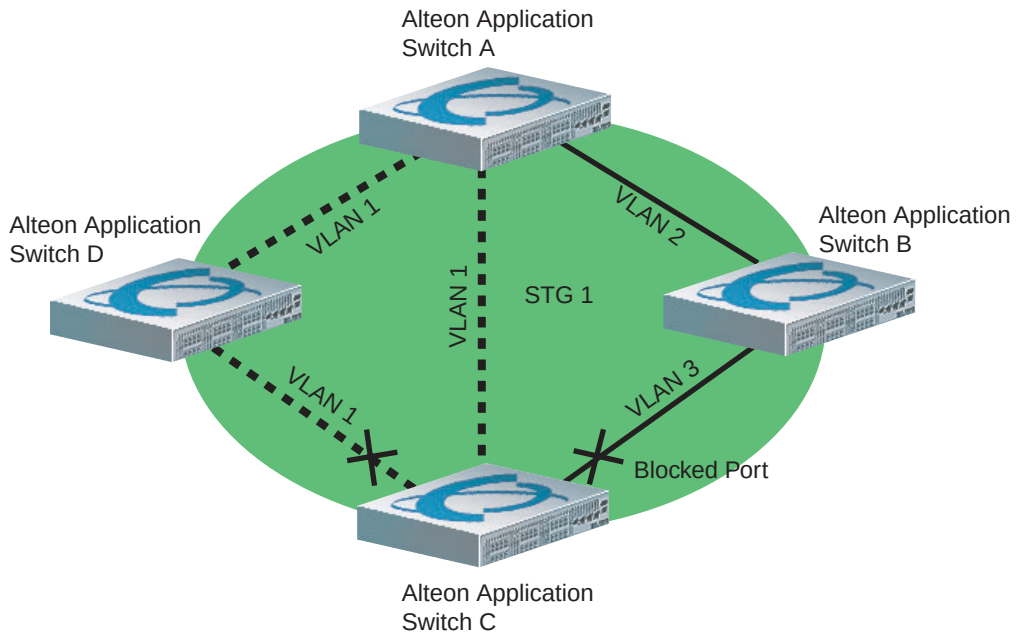


Figure 4-2 VLAN 3 Isolated in a Single Spanning Tree Group

Four-Switch Topology with Multiple Spanning Trees

If multiple Spanning Trees are implemented and each VLAN is on a different Spanning Tree, elimination of logical loops will not isolate any VLAN.

Figure 4-3 shows the same four-switch topology as in Figure 4-2 on page 65, but with multiple Spanning Trees enabled. The VLANs are identified on each of the three shaded areas connecting the switches. The port numbers are shown next to each switch. The Spanning Tree Group (STG) number for each VLAN is shown at the switch.

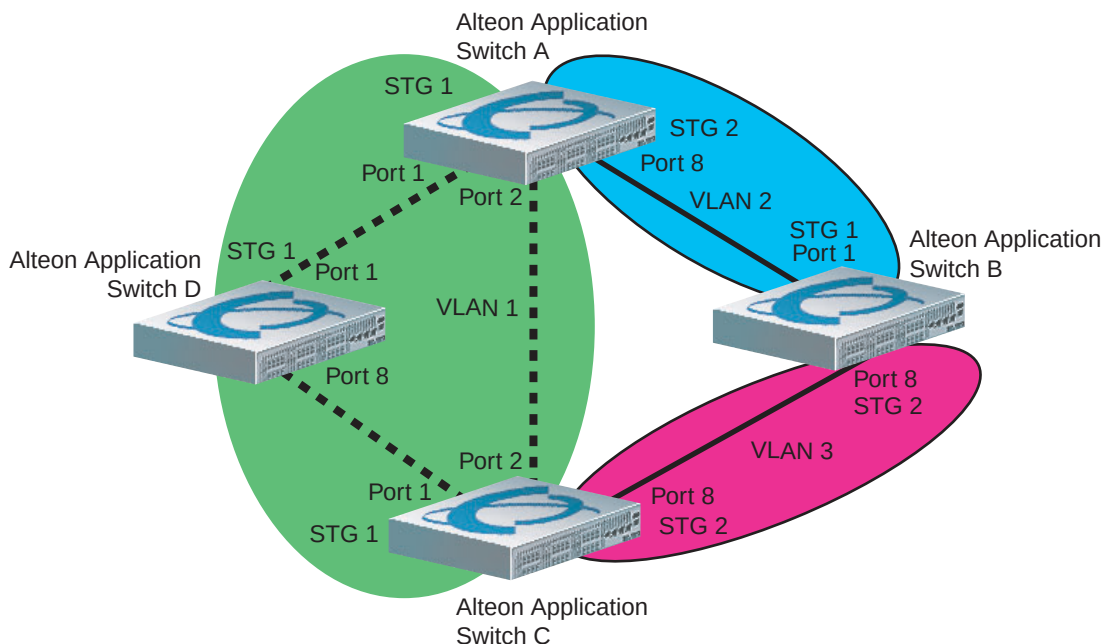


Figure 4-3 Implementing Multiple Spanning Tree Groups

Three instances of Spanning Tree are configured in the example shown in Figure 4-3. Refer to Table 4-2 on page 67 to identify the Spanning Tree group a VLAN is participating in for each switch.

Table 4-2 Multiple Spanning Tree Groups per VLAN

	VLAN 1	VLAN 2	VLAN 3
Alteon Application Switch A	Spanning Tree Group 1 Ports 1 and 2	Spanning Tree Group 2 Port 8	
Alteon Application Switch B		Spanning Tree Group 1 Port 1	Spanning Tree Group 2 Port 8
Alteon Application Switch C	Spanning Tree Group 1 Ports 1 and 2		Spanning Tree Group 2 Port 8
Alteon Application Switch D	Spanning Tree Group 1 Ports 1 and 8		

Switch-Centric Spanning Tree Protocol

In [Figure 4-3 on page 66](#), VLAN 2 is shared by application switch A and B on ports 8 and 1 respectively. Alteon Application Switch A identifies VLAN 2 in Spanning Tree group 2 and application switch B identifies VLAN 2 in Spanning Tree group 1. Spanning Tree group is switch-centric—it is used to identify the VLANs participating in the Spanning Tree groups. The Spanning Tree group ID is not transmitted in the BPDU. Each Spanning Tree decision is based on the configuration of that switch.

VLAN Participation in Spanning Tree Groups

The VLAN participation for each Spanning Tree group in [Figure 4-3 on page 66](#) is discussed in the following sections:

- **VLAN 1 Participation**
If application switch A is the root bridge, then application switch A will transmit the BPDU for VLAN 1 on ports 1 and 2. Alteon Application Switch C receives the BPDU on its port 2 and application switch D receives the BPDU on its port 1. Alteon Application Switch D will block port 8 or application switch C will block port 1 depending on the information provided in the BPDU.
- **VLAN 2 Participation**
Alteon Application Switch A, the root bridge generates another BPDU for Spanning Tree Group 2 and forwards it out from port 8. Alteon Application Switch B receives this BPDU on its port 1. Port 1 on application switch B is on VLAN 2, Spanning Tree group 1.

Because application switch B has no additional ports participating in Spanning Tree group 1, this BPDU is not be forwarded to any additional ports and application switch A remains the designated root.

■ VLAN 3 Participation

For VLAN 3 you can have application switch B or C to be the root bridge. If application switch B is the root bridge for VLAN 3, Spanning Tree group 2, then application switch B transmits the BPDU out from port 8. Alteon Application Switch C receives this BPDU on port 8 and is identified as participating in VLAN 3, Spanning Tree group 2. Since application switch C has no additional ports participating in Spanning Tree group 2, this BPDU is not forwarded to any additional ports and application switch B remains the designated root.

Configuring Multiple Spanning Tree Groups

This configuration shows how to configure the three instances of Spanning Tree groups on the application switches A, B, C, and D illustrated in [Figure 4-3 on page 66](#).

By default Spanning Trees 2-15 are empty, and Spanning Tree Group 1 contains all configured VLANs until individual VLANs are explicitly assigned to other Spanning Tree groups. You can have only one VLAN per Spanning Tree group except for Spanning Tree group 1.

1. Configure the following on application switch A:

Add port 8 to VLAN 2 and define Spanning Tree group 2 for VLAN 2.

>> # /cfg/vlan2	(Select VLAN 2 menu)
>> VLAN 2# add 8	(Add port 8)
>> VLAN 2# ../stp 2	(Select Spanning Tree Group 2)
>> Spanning Tree Group 2# add 2	(Add VLAN 2)

VLAN 2 is automatically removed from Spanning Tree Group 1.

2. Configure the following on application switch B:

Add port 1 to VLAN 2, port 8 to VLAN 3 and define Spanning Tree groups 2 for VLAN 3.

>> # /cfg/vlan2	(Select VLAN 2 menu)
>> VLAN 2# add 1	(Add port 1)
>> VLAN 2# ../vlan3	(Select VLAN 3 menu)
>> VLAN 3# add 8	(Add port 8)
>> VLAN 3# ../stp 2	(Select Spanning Tree Group 2)
>> Spanning Tree Group 2# add 3	(Add VLAN 3)

VLAN 3 is automatically removed from Spanning Tree group 1 and by default VLAN 2 remains in Spanning Tree group 1.

NOTE – Each instance of Spanning Tree group is enabled by default.

3. Configure the following on application switch C:

Add port 8 to VLAN 3 and define Spanning Tree group 3 for VLAN 3.

>> # / cfg/vlan3	<i>(Select VLAN 3 menu)</i>
>> VLAN 3# add 8	<i>(Add port 8)</i>
>> VLAN 3# ../stp 2	<i>(Select Spanning Tree Group 2)</i>
>> Spanning Tree Group 2# add 3	<i>(Add VLAN 3)</i>

VLAN 3 is automatically removed from Spanning Tree group 1 and by default VLAN 2 remains in Spanning Tree Group 1.

NOTE – Alteon Application Switch D does not require any special configuration for multiple Spanning Trees, because it configured for the default Spanning Tree group (STG 1) only.

Part 2: IP Routing

This section discusses Layer 3 switching functions. In addition to switching traffic at near line rates, the application switch can perform multi-protocol routing. This section discusses basic routing and advanced routing protocols:

- Basic Routing
- Routing Information Protocol (RIP) 1
- Border Gateway Protocol (BGP)
- Open Shortest Path First (OSPF)



CHAPTER 5

Basic IP Routing

This chapter provides configuration background and examples for using the Alteon Application Switch to perform IP routing functions. The following topics are addressed in this chapter:

- “IP Routing Benefits” on page 74
- “Routing Between IP Subnets” on page 74
- “Example of Subnet Routing” on page 77
- “Defining IP Address Ranges for the Local Route Cache” on page 81
- “Dynamic Host Configuration Protocol” on page 82

IP Routing Benefits

The Alteon Application Switch uses a combination of configurable IP switch interfaces and IP routing options. The switch IP routing capabilities provide the following benefits:

- Connects the server IP subnets to the rest of the backbone network.
- Performs server load balancing (using both Layer 3 and Layer 4 switching in combination) to server subnets that are separate from backbone subnets.
- Provides another means to invisibly introduce Jumbo frame technology into the server-switched network by automatically fragmenting UDP Jumbo frames when routing to non-Jumbo frame VLANs or subnets.
- Provides the ability to route IP traffic between multiple Virtual Local Area Networks (VLANs) configured on the switch.

Routing Between IP Subnets

The physical layout of most corporate networks has evolved over time. Classic hub/router topologies have given way to faster switched topologies, particularly now that switches are increasingly intelligent. Alteon Application Switches are intelligent and fast enough to perform routing functions on a par with wire speed Layer 2 switching.

The combination of faster routing and switching in a single device provides another service—it allows you to build versatile topologies that account for legacy configurations.

For example, consider the following topology migration:

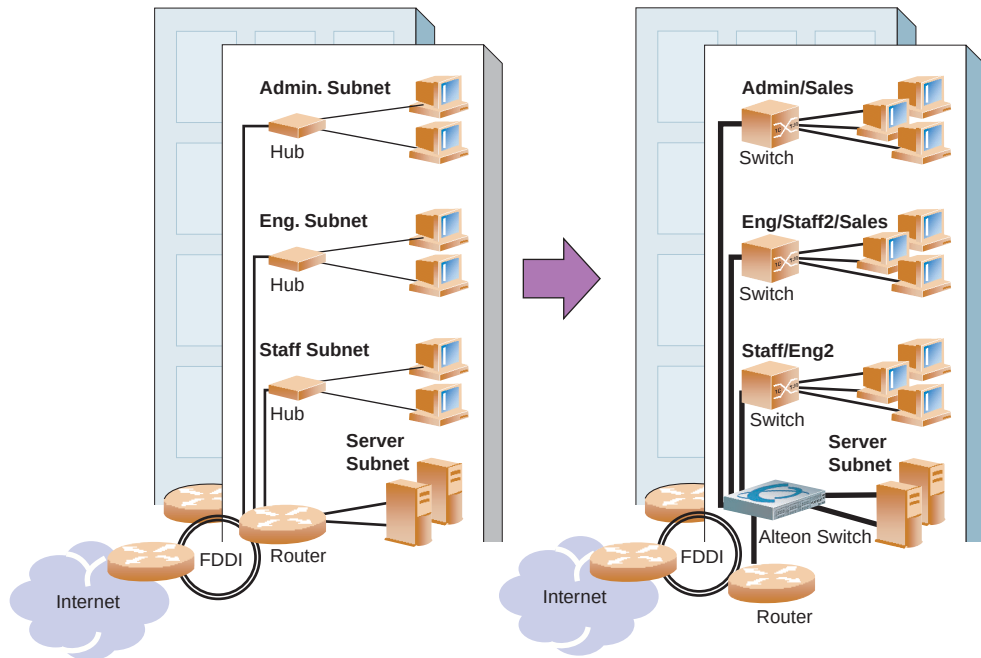


Figure 5-1 The Router Legacy Network

In this example, a corporate campus has migrated from a router-centric topology to a faster, more powerful, switch-based topology. As is often the case, the legacy of network growth and redesign has left the system with a mix of illogically distributed subnets.

This is a situation that switching alone cannot cure. Instead, the router is flooded with cross-subnet communication. This compromises efficiency in two ways:

- Routers can be slower than switches. The cross-subnet side trip from the switch to the router and back again adds two hops for the data, slowing throughput considerably.
- Traffic to the router increases, increasing congestion.

Even if every end-station could be moved to better logical subnets (a daunting task), competition for access to common server pools on different subnets still burdens the routers.

This problem is solved by using Alteon Application Switches with built-in IP routing capabilities. Cross-subnet LAN traffic can now be routed within the application switches with wire speed Layer 2 switching performance. This not only eases the load on the router but saves the network administrators from reconfiguring each and every end-station with new IP addresses.

Take a closer look at the Alteon Application Switch in the following configuration example:

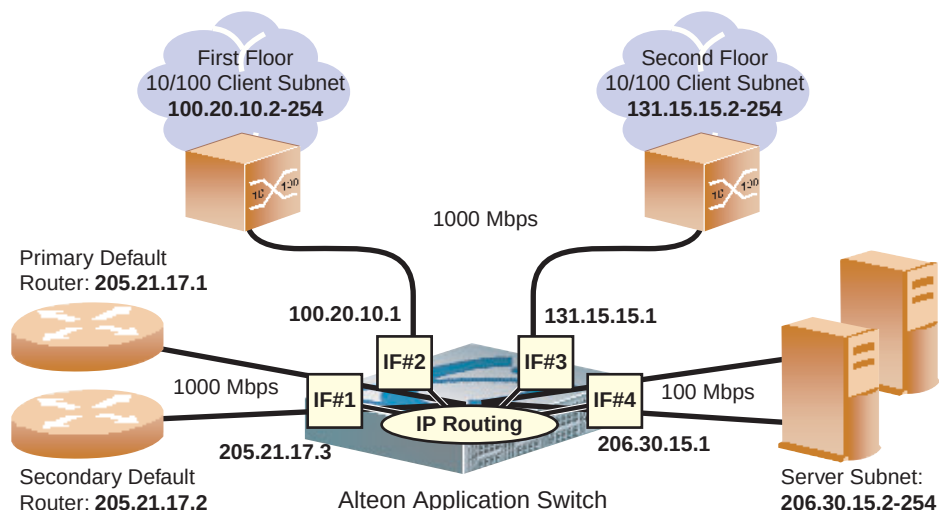


Figure 5-2 Switch-Based Routing Topology

The Alteon Application Switch connects the Gigabit Ethernet and Fast Ethernet trunks from various switched subnets throughout one building. Common servers are placed on another subnet attached to the switch. A primary and backup router are attached to the switch on yet another subnet.

Without Layer 3 IP routing on the switch, cross-subnet communication is relayed to the default gateway (in this case, the router) for the next level of routing intelligence. The router fills in the necessary address information and sends the data back to the switch, which then relays the packet to the proper destination subnet using Layer 2 switching.

With Layer 3 IP routing in place on the Alteon Application Switch, routing between different IP subnets can be accomplished entirely within the switch. This leaves the routers free to handle inbound and outbound traffic for this group of subnets.

To make implementation even easier, UDP Jumbo frame traffic is automatically fragmented to regular Ethernet frame sizes when routing to non-Jumbo frame VLANs or subnets. This automatic frame conversion allows servers to communicate using Jumbo frames, all transparently to the user.

Example of Subnet Routing

Prior to configuring, you must be connected to the switch Command Line Interface (CLI) as the administrator.

NOTE – For details about accessing and using any of the menu commands described in this example, see the *Alteon OS Command Reference*.

- 1. **Assign an IP address (or document the existing one) for each real server, router, and client workstation.**

In the example topology in [Figure 5-2 on page 76](#), the following IP addresses are used:

Table 5-1 Subnet Routing Example: IP Address Assignments

Subnet	Devices	IP Addresses
1	Primary and Secondary Default Routers	205.21.17.1 and 205.21.17.2
2	First Floor Client Workstations	100.20.10.2-254
3	Second Floor Client Workstations	131.15.15.2-254
4	Common Servers	206.30.15.2-254

- 2. **Assign an IP interface for each subnet attached to the switch.**

Since there are four IP subnets connected to the switch, four IP interfaces are needed:

Table 5-2 Subnet Routing Example: IP Interface Assignments

Interface	Devices	IP Interface Address
IF 1	Primary and Secondary Default Routers	205.21.17.3
IF 2	First Floor Client Workstations	100.20.10.1
IF 3	Second Floor Client Workstations	131.15.15.1
IF 4	Common Servers	206.30.15.1

IP interfaces are configured using the following commands at the CLI:

>> # /cfg/ip/if 1	(Select IP interface 1)
>> IP Interface 1# addr 205.21.17.3	(Assign IP address for the interface)
>> IP Interface 1# ena	(Enable IP interface 1)
>> IP Interface 1# ../if 2	(Select IP interface 2)
>> IP Interface 2# addr 100.20.10.1	(Assign IP address for the interface)
>> IP Interface 2# ena	(Enable IP interface 2)
>> IP Interface 2# ../if 3	(Select IP interface 3)
>> IP Interface 3# addr 131.15.15.1	(Assign IP address for the interface)
>> IP Interface 3# ena	(Enable IP interface 3)
>> IP Interface 3# ../if 4	(Select IP interface 4)
>> IP Interface 4# addr 206.30.15.1	(Assign IP address for the interface)
>> IP Interface 4# ena	(Enable IP interface 5)

3. Set each server and workstation's default gateway to the appropriate switch IP interface (the one in the same subnet as the server or workstation).
4. Configure the default gateways to the routers' addresses.

Configuring the default gateways allows the switch to send outbound traffic to the routers:

>> IP Interface 5# ../gw 1	(Select primary default gateway)
>> Default gateway 1# addr 205.21.17.1	(Assign IP address for primary router)
>> Default gateway 1# ena	(Enable primary default gateway)
>> Default gateway 1# ../gw 2	(Select secondary default gateway)
>> Default gateway 2# addr 205.21.17.2	(Assign address for secondary router)
>> Default gateway 2# ena	(Enable secondary default gateway)

5. Enable, apply, and verify the configuration.

>> Default gateway 2# ../fwrdr	(Select the IP Forwarding Menu)
>> IP Forwarding# on	(Turn IP forwarding on)
>> IP Forwarding# apply	(Make your changes active)
>> IP Forwarding# /cfg/ip/cur	(View current IP settings)

Examine the resulting information. If any settings are incorrect, make the appropriate changes.

6. Save your new configuration changes.

>> IP# save	(Save for restore after reboot)

Using VLANs to Segregate Broadcast Domains

In the previous example, devices that share a common IP network are all in the same broadcast domain. If you want to limit the broadcasts on your network, you could use VLANs to create distinct broadcast domains. For example, as shown in the following procedure, you could create one VLAN for the client trunks, one for the routers, and one for the servers.

In this example, you are adding to the previous configuration.

1. Determine which switch ports and IP interfaces belong to which VLANs.

The following table adds port and VLAN information:

Table 5-3 Subnet Routing Example: Optional VLAN Ports

VLAN	Devices	IP Interface	Switch Port	VLAN #
1	First Floor Client Workstations	2	1	1
	Second Floor Client Workstations	3	2	1
2	Primary Default Router	1	3	2
	Secondary Default Router	1	4	2
3	Common Servers 1	4	5	3
	Common Servers 2	4	6	3

2. Add the switch ports to their respective VLANs.

The VLANs shown in [Table 5-3](#) are configured as follows:

>> # /cfg/vlan 1	(Select VLAN 1)
>> VLAN 1# add port 1	(Add port for 1st floor to VLAN 1)
>> VLAN 1# add port 2	(Add port for 2nd floor to VLAN 1)
>> VLAN 1# ena	(Enable VLAN 1)
>> VLAN 1# ../VLAN 2	(Select VLAN 2)
>> VLAN 2# add port 3	(Add port for default router 1)
>> VLAN 2# add port 4	(Add port for default router 2)
>> VLAN 2# ena	(Enable VLAN 2)
>> VLAN 2# ../VLAN 3	(Add port for default router 3)
>> VLAN 3# add port 5	(Select VLAN 3)
>> VLAN 3# add port 6	(Select port for common server 1)
>> VLAN 3# ena	(Enable VLAN 3)

Each time you add a port to a VLAN, you may get the following prompt:

Port 4 is an untagged port and its current PVID is 1.
Confirm changing PVID from 1 to 2 [**y/n**]?

Enter **y** to set the default Port VLAN ID (PVID) for the port.

3. Add each IP interface to the appropriate VLAN.

Now that the ports are separated into three VLANs, the IP interface for each subnet must be placed in the appropriate VLAN. From [Table 5-3 on page 79](#), the settings are made as follows:

>> VLAN 3# /cfg/ip/if 1	<i>(Select IP interface 1 for def. routers)</i>
>> IP Interface 1# vlan 2	<i>(Set to VLAN 2)</i>
>> IP Interface 1# ../if 2	<i>(Select IP interface 2 for first floor)</i>
>> IP Interface 2# vlan 1	<i>(Set to VLAN 1)</i>
>> IP Interface 2# ../if 3	<i>(Select IP interface 3 for second floor)</i>
>> IP Interface 3# vlan 1	<i>(Set to VLAN 1)</i>
>> IP Interface 3# ../if 4	<i>(Select IP interface 4 for servers)</i>
>> IP Interface 4# vlan 3	<i>(Set to VLAN 3)</i>

4. Apply and verify the configuration.

>> IP Interface 5# apply	<i>(Make your changes active)</i>
>> IP Interface 5# /info/vlan	<i>(View current VLAN information)</i>
>> Information# port	<i>(View current port information)</i>

Examine the resulting information. If any settings are incorrect, make the appropriate changes.

5. Save your new configuration changes.

>> Information# save	<i>(Save for restore after reboot)</i>

Defining IP Address Ranges for the Local Route Cache

A local route cache lets you use switch resources more efficiently. The local network address and local network mask parameters (accessed via the `/cfg/ip/frwd/local/add` command) define a range of addresses that will be cached on the switch. The *local network address* is used to define the base IP address in the range that will be cached. The *local network mask* is applied to produce the range. To determine if a route should be added to the memory cache, the destination address is masked (bit-wise AND) with the local network mask and checked against the local network address.

By default, the local network address and local network mask are both set to 0.0.0.0. This produces a range that includes all Internet addresses for route caching: 0.0.0.0 through 255.255.255.255.

To limit the route cache to your local hosts, you could configure the parameters as shown in the following example:

Table 5-4 Local Routing Cache Address Ranges

Local Host Address Range	Local Network Address	Local Network Mask
0.0.0.0 - 127.255.255.255	0.0.0.0	128.0.0.0
128.0.0.0 - 128.255.255.255	128.0.0.0	128.0.0.0 or 255.0.0.0
205.32.0.0 - 205.32.255.255	205.32.0.0	255.255.0.0

NOTE – Static routes must be configured within the configured range. All other addresses that fall outside the defined range are forwarded to the default gateway.

Dynamic Host Configuration Protocol

Dynamic Host Configuration Protocol (DHCP) is a transport protocol that provides a framework for automatically assigning IP addresses and configuration information to other IP hosts or clients in a large TCP/IP network. Without DHCP, the IP address must be entered manually for each network device. DHCP allows a network administrator to distribute IP addresses from a central point and automatically send a new IP address when a device is connected to a different place in the network.

DHCP is an extension of another network IP management protocol, Bootstrap Protocol (BOOTP), with an additional capability of being able to dynamically allocate reusable network addresses and configuration parameters for client operation.

Built on the client/server model, DHCP allows hosts or clients on an IP network to obtain their configurations from a DHCP server, thereby reducing network administration. The most significant configuration the client receives from the server is its required IP address; (other optional parameters include the “generic” file name to be booted, the address of the default gateway, and so forth).

Nortel Networks DHCP relay agent eliminates the need to have DHCP/BOOTP servers on every subnet. It allows the administrator to reduce the number of DHCP servers deployed on the network and to centralize them. Without the DHCP relay agent, there must be at least one DHCP server deployed at each subnet that has hosts needing to perform the DHCP request.

DHCP Relay Agent

DHCP is described in RFC 2131, and the DHCP relay agent supported on Alteon Application switches is described in RFC 1542. DHCP uses UDP as its transport protocol. The client sends messages to the server on port 67 and the server sends messages to the client on port 68.

DHCP defines the methods through which clients can be assigned an IP address for a finite lease period and allowing reassignment of the IP address to another client later. Additionally, DHCP provides the mechanism for a client to gather other IP configuration parameters it needs to operate in the TCP/IP network.

In the DHCP environment, the Alteon Application switch acts as a relay agent. The DHCP relay feature (`/cfg/ip/bootp`) enables the switch to forward a client request for an IP address to two BOOTP servers with IP addresses that have been configured on the switch.

When a switch receives a UDP broadcast on port 67 from a DHCP client requesting an IP address, the switch acts as a proxy for the client, replacing the client source IP (SIP) and destination IP (DIP) addresses. The request is then forwarded as a UDP Unicast MAC layer message to two BOOTP servers whose IP addresses are configured on the switch. The servers respond as a UDP Unicast message back to the switch, with the default gateway and IP address for the client. The destination IP address in the server response represents the interface address on the switch that received the client request. This interface address tells the switch on which VLAN to send the server response to the client.

DHCP Relay Agent Configuration

To enable the Alteon Application switch to be the BOOTP forwarder, you need to configure the DHCP/BOOTP server IP addresses on the switch. Generally, you should configure the command on the switch IP interface closest to the client so that the DHCP server knows from which IP subnet the newly allocated IP address should come.

The following figure shows a basic DHCP network example:



Figure 5-3 DHCP Relay Agent Configuration

In Alteon Application switch implementation, there is no need for primary or secondary servers. The client request is forwarded to the BOOTP servers configured on the switch. The use of two servers provide failover redundancy. However, no health checking is supported.

Use the following commands to configure the switch as a DHCP relay agent:

```

>> # /cfg/ip/bootp
>> Bootstrap Protocol Relay# addr          (Set IP address of BOOTP server)
>> Bootstrap Protocol Relay# addr2        (Set IP address of 2nd BOOTP server)
>> Bootstrap Protocol Relay# on           (Globally turn BOOTP relay on)
>> Bootstrap Protocol Relay# off          (Globally turn BOOTP relay off)
>> Bootstrap Protocol Relay# cur           (Display current configuration)
  
```

Additionally, DHCP Relay functionality can be assigned on a per interface basis. Use the following command to enable the Relay functionality:

```

>> # /cfg/ip/if <interface number>/relay ena
  
```



CHAPTER 6

Routing Information Protocol

In a routed environment, routers communicate with one another to keep track of available routes. Routers can learn about available routes dynamically using the Routing Information Protocol (RIP). The Alteon OS software implements standard RIP for exchanging TCP/IP route information with other routers.

Distance Vector Protocol

RIP is known as a distance vector protocol. The vector is the network number and next hop, and the distance is the cost associated with the network number. RIP identifies network reachability based on cost, and cost is defined as hop count. One hop is considered to be the distance from one switch to the next which is typically 1. This cost or hop count is known as the metric.

When a switch receives a routing update that contains a new or changed destination network entry, the switch adds 1 to the metric value indicated in the update and enters the network in the routing table. The IP address of the sender is used as the next hop.

Stability

RIP version 1 was distributed in the early years of the Internet and advertised default class address without subnet masking. RIP is stable, widely supported, and easy to configure. Use RIP in stub networks and in small autonomous systems that do not have many redundant paths.

RIP includes a number of other stability features that are common to many routing protocols. For example, RIP implements the split horizon and holddown mechanisms to prevent incorrect routing information from being propagated.

RIP prevents routing loops from continuing indefinitely by implementing a limit on the number of hops allowed in a path from the source to a destination. The maximum number of hops in a path is 15. The network destination network is considered unreachable if increasing the metric value by 1 causes the metric to be 16 (that is infinity). This limits the maximum diameter of a RIP network to less than 16 hops.

Routing Updates

RIP sends routing-update messages at regular intervals and when the network topology changes. RIP uses broadcast User Datagram Protocol (UDP) data packets to exchange routing information. Each router “advertises” routing information by sending a routing information update every 30 seconds. If a router does not receive an update from another router within 90 seconds, it marks the routes served by the non-updating router as being unusable. If no update is received within 240 seconds, the router removes all routing table entries for the non-updating router.

When a router receives a routing update that includes changes to an entry, it updates its routing table to reflect the new route. The metric value for the path is increased by 1, and the sender is indicated as the next hop. RIP routers maintain only the best route (the route with the lowest metric value) to a destination.



CHAPTER 7

Border Gateway Protocol

Border Gateway Protocol (BGP) is an Internet protocol that enables routers on a network to share and advertise routing information with each other about the segments of the IP address space they can access within their network and with routers on external networks. BGP allows you to decide what is the “best” route for a packet to take from your network to a destination on another network rather than simply setting a default route from your border router(s) to your upstream provider(s). BGP is defined in RFC 1771.

Alteon Application switches can advertise their IP interfaces and virtual server IP addresses using BGP and take BGP feeds from as many as 16 BGP router peers. This allows more resilience and flexibility in balancing traffic from the Internet.

The following sections are discussed in this section:

- [“Internal Routing Versus External Routing” on page 88](#)
- [“Forming BGP Peer Routers” on page 89](#)
- [“What is a Route Map?” on page 89](#)
- [“Aggregating Routes” on page 93](#)
- [“Redistributing Routes” on page 93](#)
- [“BGP Attributes” on page 94](#)
- [“Selecting Route Paths in BGP” on page 95](#)
- [“BGP Failover Configuration” on page 96](#)
- [“Default Redistribution and Route Aggregation Example” on page 99](#)

BGP-based Global Server Load Balancing utilizes the Internet’s routing protocols to localize content delivery to the most efficient and consistent site. For more information on BGP-based GSLB, see [“Using Border Gateway Protocol for GSLB” on page 351](#).

Internal Routing Versus External Routing

To ensure effective processing of network traffic, every router on your network needs to know how to send a packet (directly or indirectly) to any other location/destination in your network. This is referred to as *internal routing* and can be done with static routes or using active, internal dynamic routing protocols, such as RIP, RIPv2, and OSPF.

Static routes, however have lower precedence than dynamic routing protocols. If the destination route is not in the route cache, then the packets are forwarded to the default gateway which may be incorrect if a dynamic routing protocol is enabled.

It is also useful to tell routers outside your network (upstream providers or *peers*) about the routes you can access in your network. External networks (those outside your own) that are under the same administrative control are referred to as *autonomous systems* (AS). Sharing of routing information between autonomous systems is known as *external routing*.

External BGP (eBGP) is used to exchange routes between different autonomous systems whereas internal BGP (iBGP) is used to exchange routes within the same autonomous system. An iBGP is a type of internal routing protocol you can use to do active routing inside your network. It also carries AS path information, which is important when you are an ISP or doing BGP transit.

NOTE – The iBGP peers must be part of a fully meshed network, as shown in Figure 7-1.

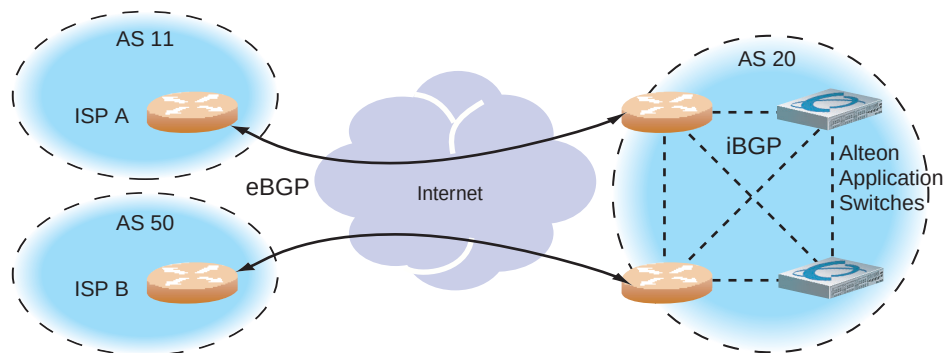


Figure 7-1 iBGP and eBGP

Typically, an AS has one or more *border routers*—peer routers that exchange routes with other ASs—and an internal routing scheme that enables routers in that AS to reach every other router and destination within that AS. When you *advertise* routes to border routers on other autonomous systems, you are effectively committing to carry data to the IP space represented in the route being advertised. For example, if you advertise 192.204.4.0/24, you are declaring that if another router sends you data destined for any address in 192.204.4.0/24, you know how to carry that data to its destination.

Forming BGP Peer Routers

Two BGP routers become peers or neighbors once you establish a TCP connection between them. For each new route, if a peer is interested in that route (for example, if a peer would like to receive your static routes and the new route is static), an update message is sent to that peer containing the new route. For each route removed from the route table, if the route has already been sent to a peer, an update message containing the route to withdraw is sent to that peer.

For each Internet host, you must be able to send a packet to that host, and that host has to have a path back to you. This means that whoever provides Internet connectivity to that host must have a path to you. Ultimately, this means that they must “hear a route” which covers the section of the IP space you are using; otherwise, you will not have connectivity to the host in question.

What is a Route Map?

A route map is used to control and modify routing information. Route maps define conditions for redistributing routes from one routing protocol to another or controlling routing information when injecting it in and out of BGP. Route maps are used by OSPF only for redistributing routes. For example, a route map is used to set a preference value for a specific route from a peer router and another preference value for all other routes learned via the same peer router. For example, the following command is used to define a route map:

```
>> # /cfg/ip/rmap 1
```

(Select a route map)

A route map allows you to match attributes, such as metric, network address, and AS number. It also allows users to overwrite the local preference metric and to append the AS number in the AS route. See “[BGP Failover Configuration](#)” on page 96.

Alteon OS allows you to configure 32 route maps. Each route map can have up to eight access lists. Each access list consists of a network filter. A network filter defines an IP address and subnet mask of the network that you want to include in the filter. [Figure 7-2](#) illustrates the relationship between route maps, access lists and network filters.

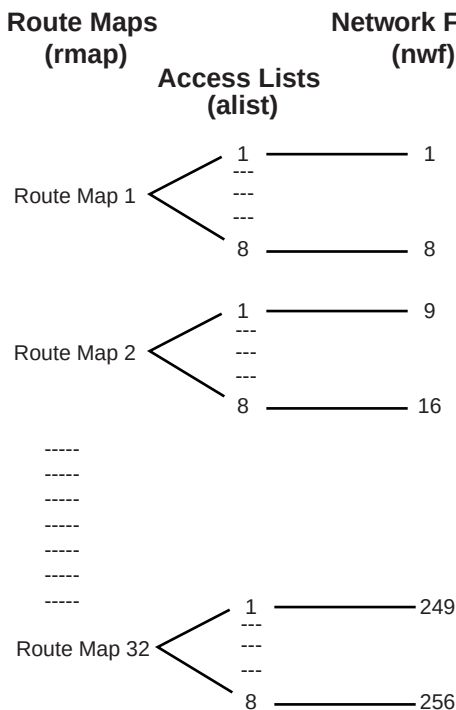


Figure 7-2 Distributing Network Filters in Access Lists and Route Maps

Incoming and Outgoing Route Maps

You can have two types of route maps: incoming and outgoing. A BGP peer router can be configured to support up to eight route maps in the incoming route map list and outgoing route map list.

If a route map is not configured in the incoming route map list, the router imports all BGP updates. If a route map is configured in the incoming route map list, the router ignores all unmatched incoming updates.

Route maps in an outgoing route map list behave similar to route maps in an incoming route map list. If a route map is not configured in the outgoing route map list, all routes are advertised or permitted. If a route map is configured in the outgoing route map list, matched routes are advertised and unmatched routes are ignored.

Precedence

You can set a priority to a route map by specifying a precedence value with the following command:

```
>> /cfg/ip/rmap/pre (Specify a precedence)
```

The smaller the value the higher the precedence. If two route maps have the same precedence value, the smaller number has higher precedence.

Configuration Overview

To configure route maps, you need to do the following:

1. Define network filter.

```
>> # /cfg/ip/nwf 1 (Specify a network filter number)
>> IP Network Filter 1# addr <IP address> (Specify network address)
>> IP Network Filter 1# mask <IP mask> (Specify network mask)
>> IP Network Filter 1# ena (Enable network filter)
```

Enter a filter number from 1 to 256. Specify the IP address and subnet mask of the network that you want to match. Enable the network filter. You can distribute up to 256 network filters among 32 route maps each containing 8 access lists.

2. (Optional) Define the criteria for the access list and enable it.

Specify the access list and associate the network filter number configured in Step 1.

```
>> # /cfg/ip/rmap 1 (Specify a route map number)
>> IP Route Map 1# alist 1 (Specify the access list number)
>> IP Access List 1# nwf 1 (Specify the network filter number)
>> IP Access List 1# metric (Define a metric)
>> IP Access List 1# action deny (Specify action for the access list)
>> IP Access List 1# ena (Enable the access list)
```

Steps 2 and 3 are optional, depending on the criteria that you want to match. In Step 2, the network filter number is used to match the subnets defined in the network filter. In Step 3, the autonomous system number is used to match the subnets. Or, you can use both (Step 2 and Step 3) criteria: access list (network filter) and access path (AS filter) to configure the route maps.

3. (Optional) Configure the attributes in the AS filter menu.

>> # cfg/ip/rmap 1/aspath 1	<i>(Specify the attributes in the filter)</i>
>> AS Filter 1# as 1	<i>(Specify the AS number)</i>
>> AS Filter 1# action deny	<i>(Specify the action for the filter)</i>
>> AS Filter 1# ena	<i>(Enable the AS filter)</i>

4. Set up the BGP attributes.

If you want to overwrite the attributes that the peer router is sending, then define the following BGP attributes:

- Specify the AS numbers that you want to prepend to a matched route and the local preference for the matched route.
- Specify the metric [Multi Exit Discriminator (MED)] for the matched route.

>> # cfg/ip/rmap 1	<i>(Specify a route map number)</i>
>> IP Route Map 1# ap	<i>(Specify the AS numbers to prepend)</i>
>> IP Route Map 1# lp	<i>(Specify the local preference)</i>
>> IP Route Map 1# med	<i>(Specify the metric)</i>

5. Enable the route map.

>> # cfg/ip/rmap 1/en	<i>(Enable the route map)</i>

6. Assign the route map to a peer router.

Select the peer router and then add the route map to the incoming route map list,

>> # cfg/ip/bgp/peer 1/addi	<i>(Add to the incoming route map)</i>

or to the outgoing route map list.

>> # cfg/ip/bgp/peer 1/addo	<i>(Add to the outgoing route map)</i>

Aggregating Routes

Aggregation is the process of combining several different routes in such a way that a single route can be advertised, which minimizes the size of the routing table. You can configure aggregate routes in BGP either by redistributing an aggregate route into BGP or by creating an aggregate entry in the BGP routing table.

When a subnet is redistributed from an Interior Gateway Protocol (IGP) into BGP, only the network route is injected into the BGP table. By default, this automatic summarization is disabled. To define the route to aggregate, use the following commands:

>> # cfg/ip/bgp	<i>(Specify BGP)</i>
>> Border Gateway Protocol# aggr 1	<i>(Specify aggregate list number)</i>
>> BGP aggr 1 # addr	<i>(Enter aggregation network address)</i>
>> BGP aggr 1 # mask	<i>(Enter aggregation network mask)</i>
>> BGP aggr 1 # ena	<i>(Enable aggregation)</i>

An example of creating a BGP aggregate route is shown in [“Default Redistribution and Route Aggregation Example” on page 99](#).

Redistributing Routes

In addition to running multiple routing protocols simultaneously, Alteon OS software can redistribute information from one routing protocol to another. For example, you can instruct the switch to use BGP to readvertise static routes. This applies to all of the IP-based routing protocols.

You can also conditionally control the redistribution of routes between routing domains by defining a method known as route maps between the two domains. For more information on route maps, see [“What is a Route Map?” on page 89](#). Redistributing routes is another way of providing policy control over whether to export OSPF routes, fixed routes, static routes, and virtual IP address routes. Default routes can be configured using the following methods:

- Import
- Originate

The router sends a default route to peers even though it does not have any default routes in its routing table.

- **Redistribute**

Default routes are either configured through the default gateway or learned via other protocols and redistributed to peer routers. If the default routes are from the default gateway, enable the static routes because default routes from the default gateway are static routes. Similarly, if the routes are learned from another routing protocol, make sure you enable that protocol for redistribution.

- **None**

BGP Attributes

The following two BGP attributes are discussed in this section: Local preference and metric (Multi-Exit Discriminator).

Local Preference Attribute

When there are multiple paths to the same destination, the local preference attribute indicates the preferred path. The path with the higher preference is preferred (the default value of the local preference attribute is 100). Unlike the weight attribute, which is only relevant to the local router, the local preference attribute is part of the routing update and is exchanged among routers in the same AS.

The local preference attribute can be set in one of two ways:

- **`/cfg/ip/bgp/pref`**

This command uses the BGP default local preference method, affecting the outbound direction only.

- **`/cfg/ip/rmap/lp`**

This command uses the route map local preference method, which affects both inbound and outbound directions.

Metric (Multi-Exit Discriminator) Attribute

This attribute is a hint to external neighbors about the preferred path into an AS when there are multiple entry points. A lower metric value is preferred over a higher metric value. The default value of the metric attribute is 0.

Unlike local preference, the metric attribute is exchanged between ASs; however, a metric attribute that comes into an AS does not leave the AS.

When an update enters the AS with a certain metric value, that value is used for decision making within the AS. When BGP sends that update to another AS, the metric is reset to 0.

Unless otherwise specified, the router compares metric attributes for paths from external neighbors that are in the same AS.

Selecting Route Paths in BGP

BGP selects only one path as the best path. It does not rely on metrics attributes to determine the best path. When the same network is learned via more than one BGP peer, BGP uses its policy for selecting the best route to that network. The BGP implementation on the Alteon Application switches uses the following criteria to select a path when the same route is received from multiple peers.

1. **Local fixed and static routes are preferred over learned routes.**
2. **With iBGP peers, routes with higher local preference values are selected.**
3. **In the case of multiple routes of equal preference, the route with lower AS path weight is selected.**

AS path weight = 128 x AS path length (number of autonomous systems transversed).

4. **In the case of equal weight and routes learned from peers that reside in the same AS, the lower metric is selected.**

NOTE – A route with a metric is preferred over a route without a metric.

5. **The lower cost to the next hop of routes is selected.**
6. **In the case of equal cost, the eBGP route is preferred over iBGP.**
7. **If all routes are from eBGP, the route with the lower router ID is selected.**

When the path is selected, BGP puts the selected path in its routing table and propagates the path to its neighbors.

BGP Failover Configuration

Use the following example to create redundant default gateways for an Alteon Application switch at a Web Host/ISP site, eliminating the possibility, should one gateway go down, that requests will be forwarded to an upstream router unknown to the switch.

As shown in Figure 7-3, the switch is connected to ISP 1 and ISP 2. The customer negotiates with both ISPs to allow the Application switch to use their peer routers as default gateways. The ISP peer routers will then need to announce themselves as default gateways to the Application switch.

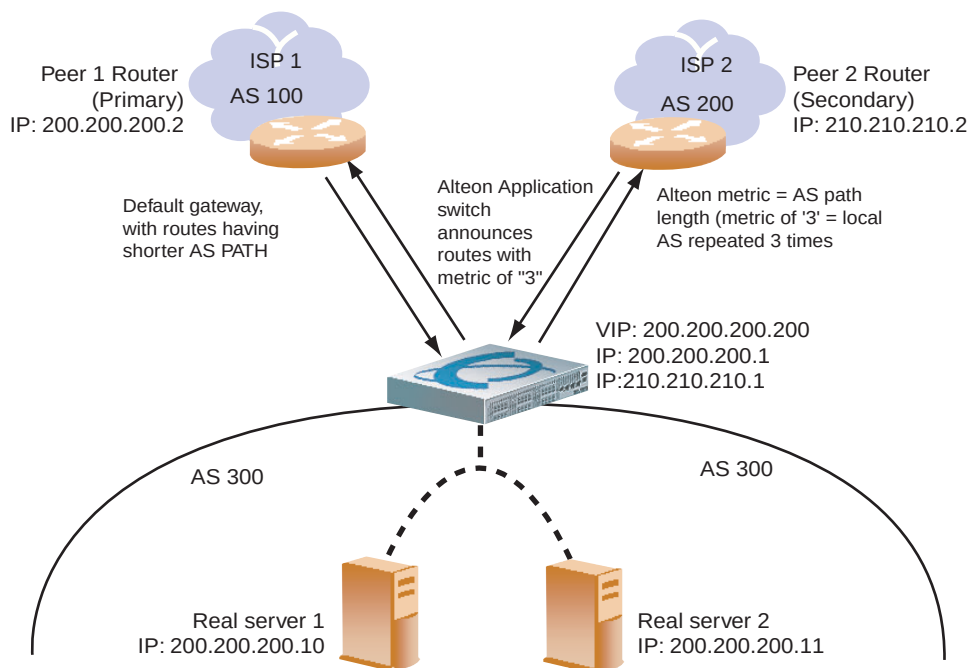


Figure 7-3 BGP Failover Configuration Example

On the Application switch, one peer router (the secondary one) is configured with a longer AS path than the other, so that the peer with the shorter AS path will be seen by the switch as the primary default gateway. ISP 2, the secondary peer, is configured with a metric of “3,” thereby appearing to the switch to be three router *hops* away.

1. Configure the switch as you normally would for Server Load Balancing (SLB).

- Assign an IP address to each of the real servers in the server pool.
- Define each real server.
- Define a real server group.
- Define a virtual server.
- Define the port configuration.

For more information about SLB configuration, refer to [Chapter 9, “Server Load Balancing.”](#)

2. Define the VLANs.

For simplicity, both default gateways are configured in the same VLAN in this example. The gateways could be in the same VLAN or different VLANs.

```
>> # /cfg/vlan 1                                (Select VLAN 1)
>> vlan 1# add <port number>                    (Add a port to the VLAN membership)
```

3. Define the IP interfaces.

The switch will need an IP interface for each default gateway to which it will be connected. Each interface will need to be placed in the appropriate VLAN. These interfaces will be used as the primary and secondary default gateways for the switch.

```
>> /cfg/ip/rearp 10                               (Set re-ARP period for interface to 10)
>> IP# metric strict                             (Set metric for default gateway)
>> IP# if 1                                       (Select default gateway interface 1)
>> IP Interface 1# ena                           (Enable switch interface 1)
>> IP Interface 1# addr 200.200.200.1            (Configure IP address of interface 1)
>> IP Interface 1# mask 255.255.255.0           (Configure IP subnet address mask)
>> IP Interface 1# broad 200.200.200.255        (Configure IP broadcast address)
>> IP Interface 1# ../ip/if 2                   (Select default gateway interface 2)
>> IP Interface 2# ena                           (Enable switch interface 2)
>> IP Interface 2# addr 210.210.210.1            (Configure IP address of interface 2)
>> IP Interface 2# mask 255.255.255.0           (Configure IP subnet address mask)
>> IP Interface 2# broad 210.210.210.255        (Configure IP broadcast address)
```

4. Enable IP forwarding.

IP forwarding is enabled by default and is used for VLAN-to-VLAN (non-BGP) routing. Make sure IP forwarding is enabled if the default gateways are on different subnets or if the switch is connected to different subnets and those subnets need to communicate through the switch (which they almost always do).

```
>> /cfg/ip/frwd on                                (Enable IP forwarding)
```

NOTE – To help eliminate the possibility for a Denial of Service (DoS) attack, the forwarding of directed broadcasts is disabled by default.

5. Configure BGP peer router 1 and 2.

Peer 1 is the primary gateway router. Peer 2 is configured with a metric of “3.” The `metric` option is key to ensuring gateway traffic is directed to Peer 1, as it will make Peer 2 appear to be three router hops away from the switch. Thus, the switch should never use it unless Peer 1 goes down.

<code>>> # /cfg/ip/bgp/peer 1</code>	<i>(Select BGP peer router 1)</i>
<code>>> BGP Peer 1# ena</code>	<i>(Enable this peer configuration)</i>
<code>>> BGP Peer 1# addr 200.200.200.2</code>	<i>(Set IP address for peer router 1)</i>
<code>>> BGP Peer 1# if 200.200.200.1</code>	<i>(Set IP interface for peer router 1)</i>
<code>>> BGP Peer 1# ras 100</code>	<i>(Set remote AS number)</i>
<code>>> BGP Peer 1# ../peer 2</code>	<i>(Select BGP peer router 2)</i>
<code>>> BGP Peer 2# ena</code>	<i>(Enable this peer configuration)</i>
<code>>> BGP Peer 2# addr 210.210.210.2</code>	<i>(Set IP address for peer router 2)</i>
<code>>> BGP Peer 2# if 210.210.210.1</code>	<i>(Set IP interface for peer router 2)</i>
<code>>> BGP Peer 2# ras 200</code>	<i>(Set remote AS number)</i>
<code>>> BGP Peer 2# metric 3</code>	<i>(Set AS path length to 3 router hops)</i>

The `metric` command in the peer menu tells the Alteon Application switch to create an AS path of “3” when advertising via BGP.

6. On the switch, apply and save your configuration changes.

<code>>> BGP Peer 2# apply</code>	<i>(Make your changes active)</i>
<code>>> save</code>	<i>(Save for restore after reboot)</i>

Default Redistribution and Route Aggregation Example

This example shows you how to configure the switch to redistribute information from one routing protocol to another and create an aggregate route entry in the BGP routing table to minimize the size of the routing table.

As illustrated in [Figure 7-4](#), you have two peer routers: an internal and an external peer router. Configure the Alteon Application switch to redistribute the default routes from AS 200 to AS 135. At the same time, configure for route aggregation to allow you to condense the number of routes traversing from AS 135 to AS 200.

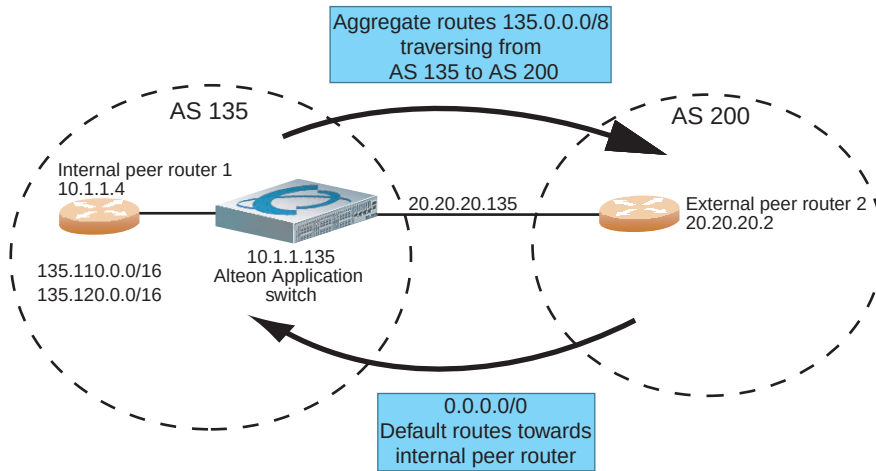


Figure 7-4 Route Aggregation and Default Route Redistribution

1. **Configure the IP interface.**
2. **Configure the AS number (AS 135) and router ID number (10.1.1.135).**

```
>> # /cfg/ip                                     (Select IP)
>> IP# as 135                                     (Specify an AS number)
>> IP# rtrid 0.0.0.1                             (Specify a router ID number)
```

3. Configure internal peer router 1 and external peer router 2.

>> # /cfg/ip/bgp/peer 1	(Select internal peer router 1)
>> BGP Peer 1# ena	(Enable this peer configuration)
>> BGP Peer 1# addr 10.1.1.4	(Set IP address for peer router 1)
>> BGP Peer 1# ras 135	(Set remote AS number)
>> BGP Peer 1# ../peer 2	(Select external peer router 2)
>> BGP Peer 2# ena	(Enable this peer configuration)
>> BGP Peer 2# addr 20.20.20.2	(Set IP address for peer router 2)
>> BGP Peer 2# ras 200	(Set remote AS number)

4. Configure redistribution for Peer 1.

>> # /cfg/ip/bgp/peer 1/redist	(Select redistribute)
>> BGP Peer 1# default redistribute	(Set default to redistribute)
>> BGP Peer 1# fixed ena	(Enable fixed routes)

5. Configure aggregation policy control.

Configure the routes that you want aggregated.

>> # /cfg/ip/bgp/aggr 1	(Set aggregation number)
>> BGP aggr 1# addr 135.0.0.0	(Add IP address to aggregate 1)
>> BGP Peer 1# mask 255.0.0.0	(Add IP mask to aggregate 1)



CHAPTER 8

OSPF

Alteon OS supports the Open Shortest Path First (OSPF) routing protocol. The Alteon OS implementation conforms to the OSPF version 2 specifications detailed in Internet RFC 1583. The following sections discuss OSPF support for the Alteon Application Switches:

- [“OSPF Overview” on page 102](#). This section provides information on OSPF concepts, such as types of OSPF areas, types of routing devices, neighbors, adjacencies, link state database, authentication, and internal versus external routing.
- [“OSPF Implementation in Alteon OS” on page 107](#). This section describes how OSPF is implemented in Alteon OS, such as configuration parameters, electing the designated router, summarizing routes, defining route maps and so forth.
- [“OSPF Configuration Examples” on page 117](#). This section provides step-by-step instructions on configuring four different configuration examples:
 - Creating a simple OSPF domain
 - Creating virtual links
 - Summarizing routes
 - Creating host routes

OSPF Overview

OSPF is designed for routing traffic within a single IP domain called an Autonomous System (AS). The AS can be divided into smaller logical units known as *areas*.

All routing devices maintain link information in their own Link State Database (LSDB). The LSDB for all routing devices within an area is identical but is not exchanged between different areas. Only routing updates are exchanged between areas, thereby significantly reducing the overhead for maintaining routing information on a large, dynamic network.

The following sections describe key OSPF concepts.

Types of OSPF Areas

An AS can be broken into logical units known as *areas*. In any AS with multiple areas, one area must be designated as area 0, known as the *backbone*. The backbone acts as the central OSPF area. All other areas in the AS must be connected to the backbone. Areas inject summary routing information into the backbone, which then distributes it to other areas as needed.

As shown in [Figure 8-1](#), OSPF defines the following types of areas:

- **Stub Area**—an area that is connected to only one other area. External route information is not distributed into stub areas.
- **Not-So-Stubby-Area (NSSA)**—similar to a stub area with additional capabilities. Routes originating from within the NSSA can be propagated to adjacent transit and backbone areas. External routes from outside the AS can be advertised within the NSSA but are not distributed into other areas.

- **Transit Area**—an area that allows area summary information to be exchanged between routing devices. The backbone (area 0), any area that contains a virtual link to connect two areas, and any area that is not a stub area or an NSSA are considered transit areas.

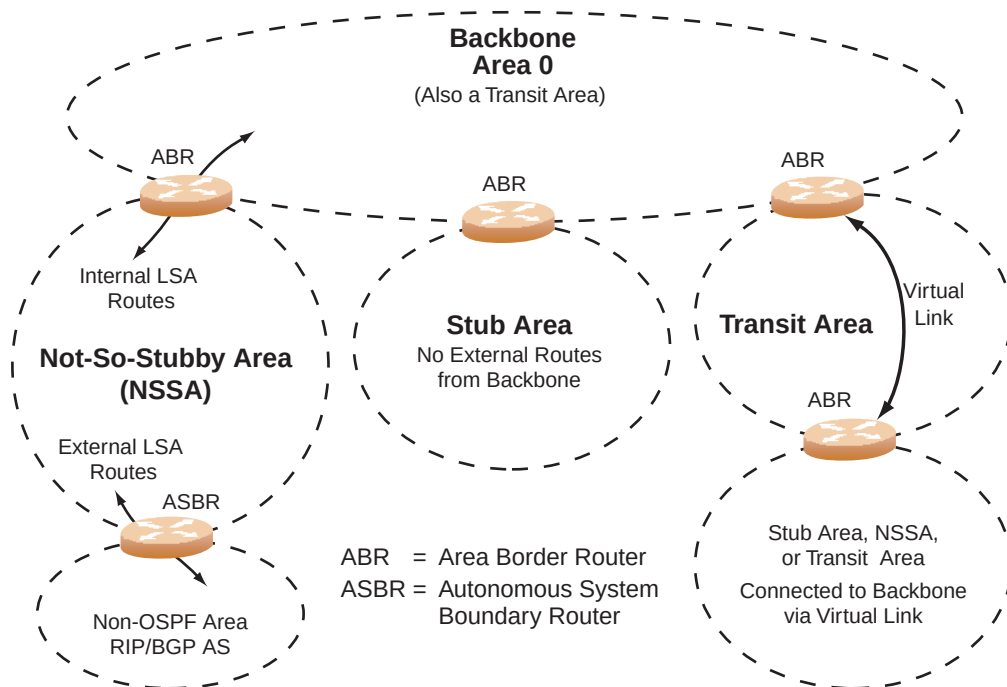


Figure 8-1 OSPF Area Types

Types of OSPF Routing Devices

As shown in [Figure 8-2](#), OSPF uses the following types of routing devices:

- Internal Router (IR)—a router that has all of its interfaces within the same area. IRs maintain LSDBs identical to those of other routing devices within the local area.
- Area Border Router (ABR)—a router that has interfaces in multiple areas. ABRs maintain one LSDB for each connected area and disseminate routing information between areas.
- Autonomous System Boundary Router (ASBR)—a router that acts as a gateway between the OSPF domain and non-OSPF domains, such as RIP, BGP, and static routes.

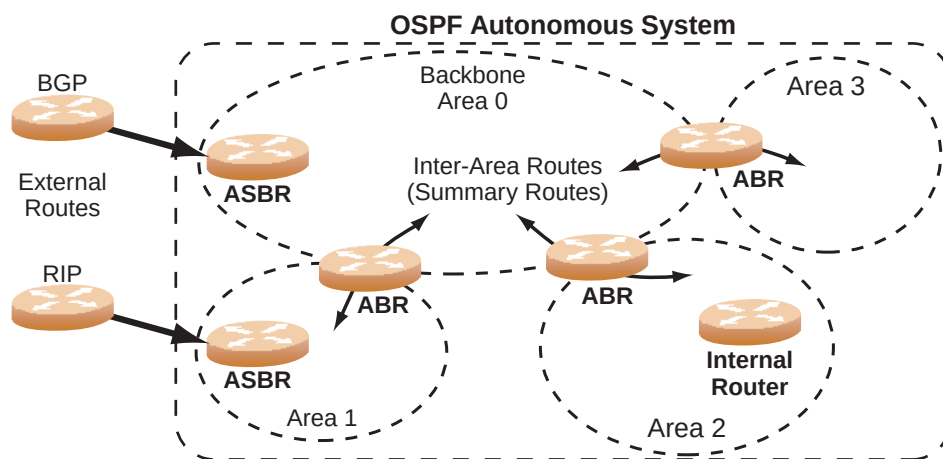


Figure 8-2 OSPF Domain and an Autonomous System

Neighbors and Adjacencies

In areas with two or more routing devices, *neighbors* and *adjacencies* are formed.

Neighbors are routing devices that maintain information about each others' health. To establish neighbor relationships, routing devices periodically send hello packets on each of their interfaces. All routing devices that share a common network segment, appear in the same area, and have the same health parameters (hello and dead intervals) and authentication parameters respond to each other's hello packets and become neighbors. Neighbors continue to send periodic hello packets to advertise their health to neighbors. In turn, they listen to hello packets to determine the health of their neighbors and to establish contact with new neighbors.

The hello process is used for electing one of the neighbors as the area's Designated Router (DR) and one as the area's Backup Designated Router (BDR). The DR is adjacent to all other neighbors and acts as the central contact for database exchanges. Each neighbor sends its database information to the DR, which relays the information to the other neighbors.

The BDR is adjacent to all other neighbors (including the DR). Each neighbor sends its database information to the BDR just as with the DR, but the BDR merely stores this data and does not distribute it. If the DR fails, the BDR will take over the task of distributing database information to the other neighbors.

The Link-State Database

OSPF is a link-state routing protocol. A *link* represents an interface (or routable path) from the routing device. By establishing an adjacency with the DR, each routing device in an OSPF area maintains an identical Link-State Database (LSDB) describing the network topology for its area.

Each routing device transmits a Link-State Advertisement (LSA) on each of its interfaces. LSAs are entered into the LSDB of each routing device. OSPF uses *flooding* to distribute LSAs between routing devices.

When LSAs result in changes to the routing device's LSDB, the routing device forwards the changes to the adjacent neighbors (the DR and BDR) for distribution to the other neighbors.

OSPF routing updates occur only when changes occur, instead of periodically. For each new route, if an adjacency is interested in that route (for example, if configured to receive static routes and the new route is indeed static), an update message containing the new route is sent to the adjacency. For each route removed from the route table, if the route has already been sent to an adjacency, an update message containing the route to withdraw is sent.

The Shortest Path First Tree

The routing devices use a link-state algorithm (Dijkstra's algorithm) to calculate the shortest path to all known destinations, based on the cumulative *cost* required to reach the destination.

The cost of an individual interface in OSPF is an indication of the overhead required to send packets across it. The cost is inversely proportional to the bandwidth of the interface. A lower cost indicates a higher bandwidth.

Internal Versus External Routing

To ensure effective processing of network traffic, every routing device on your network needs to know how to send a packet (directly or indirectly) to any other location/destination in your network. This is referred to as *internal routing* and can be done with static routes or using active internal routing protocols, such as OSPF, RIP, or RIPv2.

It is also useful to tell routers outside your network (upstream providers or *peers*) about the routes you have access to in your network. Sharing of routing information between autonomous systems is known as *external routing*.

Typically, an AS will have one or more border routers (peer routers that exchange routes with other OSPF networks) as well as an internal routing system enabling every router in that AS to reach every other router and destination within that AS.

When a routing device *advertises* routes to boundary routers on other autonomous systems, it is effectively committing to carry data to the IP space represented in the route being advertised. For example, if the routing device advertises 192.204.4.0/24, it is declaring that if another router sends data destined for any address in the 192.204.4.0/24 range, it will carry that data to its destination.

OSPF Implementation in Alteon OS

Alteon OS supports a single instance of OSPF and up to 4 K routes on the network. The following sections describe OSPF implementation in Alteon OS:

- “Configurable Parameters” on page 107
- “Defining Areas” on page 108
- “Interface Cost” on page 110
- “Electing the Designated Router and Backup” on page 110
- “Summarizing Routes” on page 110
- “Default Routes” on page 111
- “Virtual Links” on page 112
- “Router ID” on page 113
- “Authentication” on page 113
- “Host Routes for Load Balancing” on page 116

Configurable Parameters

In Alteon OS, OSPF parameters can be configured through the Command Line Interface (CLI), Alteon OS Browser-Based Interface (BBI) for Alteon Application Switches, or through SNMP. For more information, see [Chapter 1, “Accessing the Switch.”](#)

The CLI supports the following parameters: interface output cost, interface priority, dead and hello intervals, retransmission interval, and interface transmit delay.

In addition to the above parameters, you can also specify the following:

- Shortest Path First (SPF) interval—Time interval between successive calculations of the shortest path tree using the Dijkstra’s algorithm.
- Stub area metric—A stub area can be configured to send a numeric metric value such that all routes received via that stub area carry the configured metric to potentially influence routing decisions.
- Default routes—Default routes with weight metrics can be manually injected into transit areas. This helps establish a preferred route when multiple routing devices exist between two areas. It also helps route traffic to external networks.

Defining Areas

If you are configuring multiple areas in your OSPF domain, one of the areas must be designated as area 0, known as the *backbone*. The backbone is the central OSPF area and is usually physically connected to all other areas. The areas inject routing information into the backbone which, in turn, disseminates the information into other areas.

Since the backbone connects the areas in your network, it must be a contiguous area. If the backbone is partitioned (possibly as a result of joining separate OSPF networks), parts of the AS will be unreachable, and you will need to configure *virtual links* to reconnect the partitioned areas (see “[Virtual Links](#)” on page 112).

Up to three OSPF areas can be connected to the Alteon Application Switch with Alteon OS software. To configure an area, the OSPF number must be defined and then attached to a network interface on the switch. The full process is explained in the following sections.

An OSPF area is defined by assigning **two** pieces of information—an *area index* and an *area ID*. The command to define an OSPF area is as follows:

```
>> # /cfg/ip/ospf/aindex <area index>/areaid <n.n.n.n>
```

NOTE – The `aindex` option above is an arbitrary index used only on the switch and does not represent the actual OSPF area number. The actual OSPF area number is defined in the `areaid` portion of the command as explained in the following sections.

Assigning the Area Index

The `aindex <area index>` option is actually just an arbitrary index (0-2) used only by the switch. This index does not necessarily represent the OSPF area number, though for configuration simplicity, it should where possible.

For example, both of the following sets of commands define OSPF area 0 (the backbone) and area 1 because that information is held in the area ID portion of the command. However, the first set of commands is easier to maintain because the arbitrary area indexes agree with the area IDs:

- Area index and area ID agree

```
/cfg/ip/ospf/aindex 0/areaid 0.0.0.0 (Use index 0 to set area 0 in ID octet format)
/cfg/ip/ospf/aindex 1/areaid 0.0.0.1 (Use index 1 to set area 1 in ID octet format)
```

- Area index set to an arbitrary value

```
/cfg/ip/ospf/aindex 1/areaid 0.0.0.0 (Use index 1 to set area 0 in ID octet format)
/cfg/ip/ospf/aindex 2/areaid 0.0.0.1 (Use index 2 to set area 1 in ID octet format)
```

Using the Area ID to Assign the OSPF Area Number

The OSPF area number is defined in the `areaid <IP address>` option. The octet format is used in order to be compatible with two different systems of notation used by other OSPF network vendors. There are two valid ways to designate an area ID:

- **Placing the area number in the last octet (0.0.0.*n*)**

Most common OSPF vendors express the area ID number as a single number. For example, the Cisco IOS-based router command “network 1.1.1.0 0.0.0.255 area 1” defines the area number simply as “area 1.” On the application switch, using the last octet in the area ID, “area 1” is equivalent to “areaid 0.0.0.1”.

- **Multi-octet (*IP address*)**

Some OSPF vendors express the area ID number in multi-octet format. For example, “area 2.2.2.2” represents OSPF area 2 and can be specified directly on the Alteon Application Switch as “areaid 2.2.2.2”.

NOTE – Although both types of area ID formats are supported, be sure that the area IDs are in the same format throughout an area.

Attaching an Area to a Network

Once an OSPF area has been defined, it must be associated with a network. To attach the area to a network, you must assign the OSPF area index to an IP interface that participates in the area. The format for the command is as follows:

```
>> # /cfg/ip/ospf/if <interface number>/aindex <area index>
```

For example, the following commands could be used to configure IP interface 14 for a presence on the 10.10.10.1/24 network, to define OSPF area 1, and to attach the area to the network:

>> # /cfg/ip/if 14	(Select menu for IP interface 14)
>> IP Interface 14# addr 10.10.10.1	(Define IP address on backbone network)
>> IP Interface 14# mask 255.255.255.0	(Define IP mask on backbone)
>> IP Interface 14# ena	(Enable IP interface 14)
>> IP Interface 14# ../ospf/aindex 1	(Select menu for area index 1)
>> OSPF Area (index) 1 # areaid 0.0.0.1	(Define area ID as OSPF area 1)
>> OSPF Area (index) 1 # ena	(Enable area index 1)
>> OSPF Area (index) 1 # ../if 14	(Select OSPF menu for interface 14)
>> OSPF Interface 14# aindex 1	(Attach area to network on interface 14)
>> OSPF Interface 14# enable	(Enable interface 14 for area index 1)

Interface Cost

The OSPF link-state algorithm (Dijkstra's algorithm) places each routing device at the root of a tree and determines the cumulative *cost* required to reach each destination. Usually, the cost is inversely proportional to the bandwidth of the interface. Low cost indicates high bandwidth. You can manually enter the cost for the output route with the following command:

```
>> # /cfg/ip/ospf/if <OSPF interface number>/cost <cost value (1-65535)>
```

Electing the Designated Router and Backup

In any area with more than two routing devices, a Designated Router (DR) is elected as the central contact for database exchanges among neighbors, and a Backup Designated Router (BDR) is elected in case the DR fails.

DR and BDR elections are made through the hello process. The election can be influenced by assigning a priority value to the OSPF interfaces on the Alteon Application Switch. The command is as follows:

```
>> # /cfg/ip/ospf/if <OSPF interface number>/prio <priority value (0-127)>
```

A priority value of 127 is the highest, and 1 is the lowest. A priority value of 0 specifies that the interface cannot be used as a DR or BDR. In case of a tie, the routing device with the lowest router ID wins.

Summarizing Routes

Route summarization condenses routing information. Without summarization, each routing device in an OSPF network would retain a route to every subnet in the network. With summarization, routing devices can reduce some sets of routes to a single advertisement, reducing both the load on the routing device and the perceived complexity of the network. The importance of route summarization increases with network size.

Summary routes can be defined for up to 16 IP address ranges using the following command:

```
>> # /cfg/ip/ospf/range <range number>/addr <IP address>/mask  
<mask>
```

where *<range number>* is a number 1 to 16, *<IP address>* is the base IP address for the range, and *<mask>* is the IP address mask for the range. For a detailed configuration example, see [“Example 3: Summarizing Routes” on page 124](#).

Default Routes

When an OSPF routing device encounters traffic for a destination address it does not recognize, it forwards that traffic along the *default route*. Typically, the default route leads upstream toward the backbone until it reaches the intended area or an external router.

Each Alteon Application Switch acting as an ABR automatically inserts a default route into each attached area. In simple OSPF stub areas or NSSAs with only one ABR leading upstream (see Area 1 in [Figure 8-3](#)), any traffic for IP address destinations outside the area is forwarded to the switch's IP interface, and then into the connected transit area (usually the backbone). Since this is automatic, no further configuration is required for such areas.

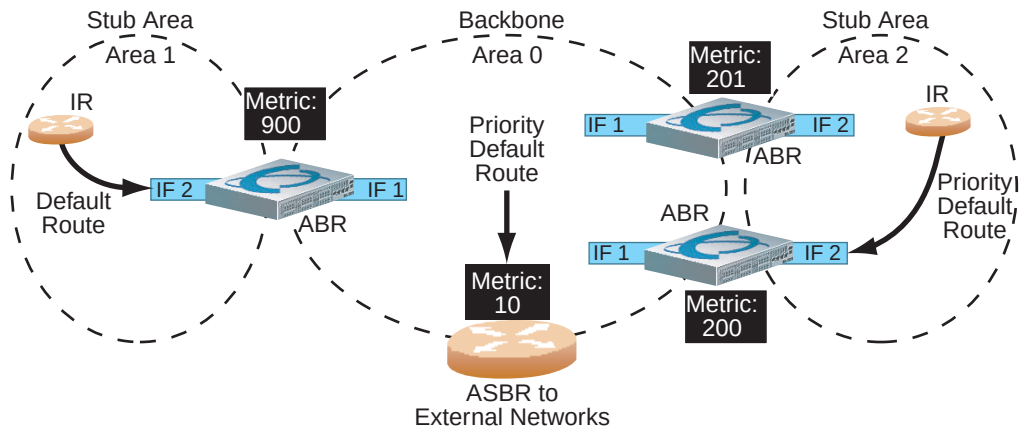


Figure 8-3 Injecting Default Routes

In more complex OSPF areas with multiple ABRs or ASBRs (such as area 0 and area 2 in [Figure 8-3](#)), there are multiple routes leading from the area. In such areas, traffic for unrecognized destinations cannot tell which route leads upstream without further configuration.

To resolve the situation and select one default route among multiple choices in an area, you can manually configure a metric value on each ABR. The metric assigns a priority to the ABR for its selection as the priority default route in an area. The following command is used for setting the metric value:

```
>> # /cfg/ip/ospf/default <metric value> <metric type (1 or 2)>
```

where *<metric value>* sets the priority for choosing this switch for default route. The value *none* sets no default and 1 sets the highest priority for default route. Metric type determines the method for influencing routing decisions for external routes.

To clear a default route metric from the switch, use the following command:

```
>> # /cfg/ip/ospf/default none
```

Virtual Links

Usually, all areas in an OSPF AS are physically connected to the backbone. In some cases where this is not possible, you can use a *virtual link*. Virtual links are created to connect one area to the backbone through another non-backbone area (see [Figure 8-1 on page 103](#)).

The area which contains a virtual link must be a transit area and have full routing information. Virtual links cannot be configured inside a stub area or NSSA. The area type must be defined as `transit` using the following command:

```
>> # /cfg/ip/ospf/aindex <area index>/type transit
```

The virtual link must be configured on the routing devices at each endpoint of the virtual link, though they may traverse multiple routing devices. To configure an Alteon Application Switch as one endpoint of a virtual link, use the following command:

```
>> # /cfg/ip/ospf/virt <link number>/aindex <area index>/nbr <router ID>
```

where *<link number>* is a value between 1 and 3, *<area index>* is the OSPF area index of the transit area, and *<router ID>* is the IP address of the virtual neighbor (nbr), the routing device at the target endpoint. Another router ID is needed when configuring a virtual link in the other direction. To provide the Alteon Application Switch with a router ID, see the following section [Router ID](#).

For a detailed configuration example on Virtual Links, see [“Example 2: Virtual Links” on page 120](#).

Router ID

Routing devices in OSPF areas are identified by a router ID. The router ID is expressed in IP address format. The IP address of the router ID is not required to be included in any IP interface range or in any OSPF area.

The router ID can be configured in one of the following two ways:

- Dynamically—OSPF protocol configures the lowest IP interface IP address as the router ID. This is the default.
- Statically—Use the following command to manually configure the router ID:

```
>> # /cfg/ip/rtrid <IP address>
```

- To modify the router ID from static to dynamic, set the router ID to 0.0.0.0, save the configuration, and reboot the Alteon Application Switch. To view the router ID, enter:

```
>> # /info/ospf/gen
```

Authentication

OSPF protocol exchanges can be authenticated so that only trusted routing devices can participate. This ensures less processing on routing devices that are not listening to OSPF packets.

OSPF allows packet authentication and uses IP multicast when sending and receiving packets. Routers participate in routing domains based on predefined passwords. Alteon OS supports simple password (type 1 plain text passwords) and MD5 cryptographic authentication. This type of authentication allows a password to be configured per area.

Figure 8-4 shows authentication configured for area 0 with the password test. Simple authentication is also configured for the virtual link between area 2 and area 0. Area 1 is not configured for OSPF authentication.

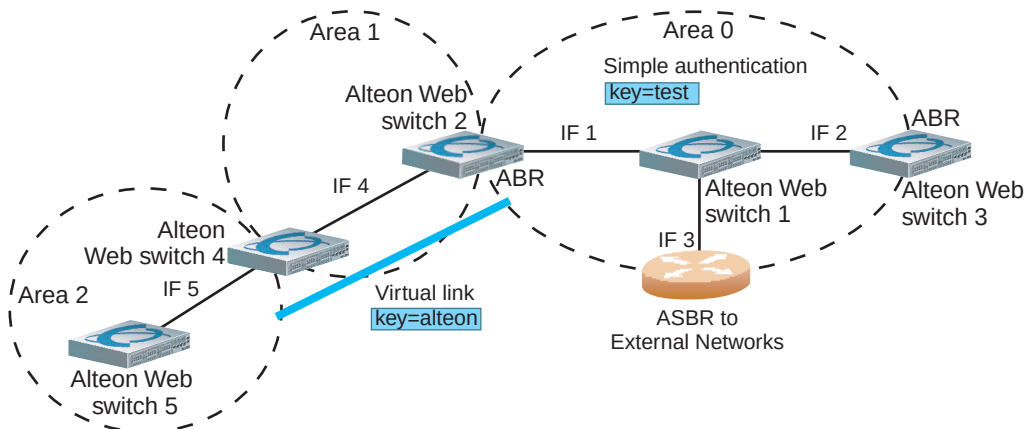


Figure 8-4 OSPF Authentication

To configure simple plain text OSPF passwords on the Alteon switches shown in [Figure 8-4](#) use the following commands:

1. **Enable OSPF authentication for Area 0 on Alteon switches 1, 2, and 3.**

```
>> # /cfg/ip/ospf/aindex 0/auth password
                                     (Turn on OSPF password authentication)
```

2. **Configure a simple text password up to eight characters for each OSPF IP interface in Area 0 on Alteon switches 1, 2, and 3.**

```
>> # /cfg/ip/ospf/if 1
>> OSPF Interface 1 # key test
>> OSPF Interface 1 # ../if 2
>> OSPF Interface 2 # key test
>> OSPF Interface 1 # ../if 3
>> OSPF Interface 3 # key test
```

3. **Enable OSPF authentication for Area 2 on Alteon Application Switch 4.**

```
>> # /cfg/ip/ospf/aindex 2/auth password
                                     (Turn on OSPF password authentication)
```

4. **Configure a simple text password up to eight characters for the virtual link between Area 2 and Area 0 on Alteon switches 2 and 4.**

```
>> # /cfg/ip/ospf/virt 1/key alteon
```

Use the following commands to configure MD5 authentication on the Alteon switches shown in [Figure 8-4](#):

1. **Enable OSPF MD5 authentication for Area 0 on Alteon switches 1, 2, and 3.**

```
>> # /cfg/ip/ospf/aindex 0/auth md5      (Turn on MD5 authentication)
```

2. **Configure MD5 key ID for Area 0 on Alteon switches 1, 2, and 3.**

```
>> # /cfg/ip/ospf/md5key 1/key test
```

3. Assign MD5 key ID to OSPF interfaces on Alteon Application switches 1, 2, and 3.

```
>> # /cfg/ip/ospf/if 1
>> OSPF Interface 1 # mdkey 1
>> OSPF Interface 1 # ../if 2
>> OSPF Interface 2 # mdkey 1
>> OSPF Interface 1 # ../if 3
>> OSPF Interface 3 # mdkey 1
```

4. Enable OSPF MD5 authentication for Area 2 on Alteon Application Switch 4.

```
>> # /cfg/ip/ospf/aindex 2/auth md5
```

5. Configure MD5 key for the virtual link between Area 2 and Area 0 on Alteon switches 2 and 4.

```
>> # /cfg/ip/ospf/md5key 2/key alteon
```

6. Assign MD5 key ID to OSPF virtual link on Alteon switches 2 and 4.

```
>> # /cfg/ip/ospf/virt 1/mdkey 2
```

Host Routes for Load Balancing

Alteon OS implementation of OSPF includes host routes. Host routes are used for advertising network device IP addresses to external networks, accomplishing the following goals:

- **Server Load Balancing (SLB) within OSPF**

Host routes advertise virtual server IP addresses to external networks. This allows standard SLB between the Alteon Application Switch and the server pools in an OSPF environment. For more information on SLB, see [Chapter 9, “Server Load Balancing”](#) and your *Alteon OS Command Reference*.

- **ABR Load Sharing**

As a second form of load balancing, host routes can be used for dividing OSPF traffic among multiple ABRs. To accomplish this, each application switch provides identical services but advertises a host route for a different virtual server IP address to the external network. If each virtual server IP address serves a different and equal portion of the external world, incoming traffic from the upstream router should be split evenly among ABRs.

- **ABR Failover**

Complementing ABR load sharing, identical host routes can be configured on each ABR. These host routes can be given different costs so that a different ABR is selected as the preferred route for each virtual server and the others are available as backups for failover purposes.

If redundant routes via multiple routing processes (such as OSPF, RIP, BGP, or static routes) exist on your network, the application switch defaults to the OSPF-derived route.

For a configuration example, see [“Example 4: Host Routes” on page 127](#).

OSPF Features Not Supported in This Release

The following OSPF features are not supported in this release:

- Summarizing external routes
- Filtering OSPF routes
- Configuring equal cost route load balancing
- Using OSPF to forward multicast routes
- Configuring OSPF on non-broadcast multi-access networks (such as frame relay, X.25, and ATM)

OSPF Configuration Examples

A summary of the basic steps for configuring OSPF on the Alteon Application Switch is listed here. Detailed instructions for each of the steps is covered in the following sections:

1. **Configure IP interfaces.**

One IP interface is required for each desired network (range of IP addresses) being assigned to an OSPF area on the Alteon Application Switch.

2. **(Optional) Configure the router ID.**

The router ID is required only when configuring virtual links on the Alteon Application Switch.

3. **Enable OSPF on the switch.**

4. **Define the OSPF areas.**

5. **Configure OSPF interface parameters.**

IP interfaces are used for attaching networks to the various areas.

6. **(Optional) Configure route summarization between OSPF areas.**

7. **(Optional) Configure virtual links.**

8. **(Optional) Configure host routes.**

Example 1: Simple OSPF Domain

In this example, two OSPF areas are defined—one area is the backbone and the other is a stub area. A stub area does not allow advertisements of external routes, thus reducing the size of the database. Instead, a default summary route of IP address 0.0.0.0 is automatically inserted into the stub area. Any traffic for IP address destinations outside the stub area will be forwarded to the stub area's IP interface, and then into the backbone.

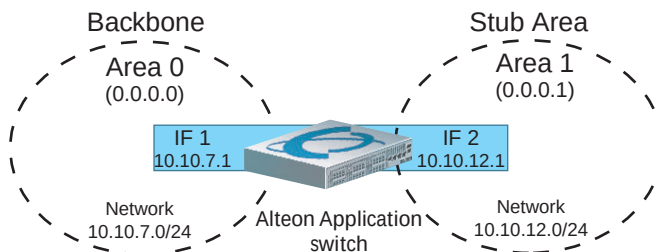


Figure 8-5 A Simple OSPF Domain

Follow this procedure to configure OSPF support as shown in [Figure 8-5](#):

1. Configure IP interfaces on each network that will be attached to OSPF areas.

In this example, two IP interfaces are needed: one for the backbone network on 10.10.7.0/24 and one for the stub area network on 10.10.12.0/24.

<code>>> # /cfg/ip/if 1</code>	<i>(Select menu for IP interface 1)</i>
<code>>> IP Interface 1 # addr 10.10.7.1</code>	<i>(Set IP address on backbone network)</i>
<code>>> IP Interface 1 # mask 255.255.255.0</code>	<i>(Set IP mask on backbone network)</i>
<code>>> IP Interface 1 # broad 10.10.7.255</code>	<i>(Set the broadcast address)</i>
<code>>> IP Interface 1 # enable</code>	<i>(Enable IP interface 1)</i>
<code>>> IP Interface 1 # ../if 2</code>	<i>(Select menu for IP interface 2)</i>
<code>>> IP Interface 2 # addr 10.10.12.1</code>	<i>(Set IP address on stub area network)</i>
<code>>> IP Interface 2 # mask 255.255.255.0</code>	<i>(Set IP mask on stub area network)</i>
<code>>> IP Interface 2 # broad 10.10.12.255</code>	<i>(Set the broadcast address)</i>
<code>>> IP Interface 2 # enable</code>	<i>(Enable IP interface 2)</i>

2. Enable OSPF.

<code>>> IP Interface 2 # /cfg/ip/ospf/on</code>	<i>(Enable OSPF on the Alteon switch)</i>

3. Define the backbone.

The backbone is always configured as a transit area using `areaid 0.0.0.0`.

```
>> Open Shortest Path First # aindex 0      (Select menu for area index 0)
>> OSPF Area (index) 0 # areaid 0.0.0.0    (Set the ID for backbone area 0)
>> OSPF Area (index) 0 # type transit      (Define backbone as transit type)
>> OSPF Area (index) 0 # enable           (Enable the area)
```

4. Define the stub area.

```
>> OSPF Area (index) 0 # ../aindex 1      (Select menu for area index 1)
>> OSPF Area (index) 1 # areaid 0.0.0.1    (Set the area ID for OSPF area 1)
>> OSPF Area (index) 1 # type stub        (Define area as stub type)
>> OSPF Area (index) 1 # enable           (Enable the area)
```

5. Attach the network interface to the backbone.

```
>> OSPF Area 1 # ../if 1                  (Select OSPF menu for IP interface 1)
>> OSPF Interface 1 # aindex 0            (Attach network to backbone index)
>> OSPF Interface 1 # enable              (Enable the backbone interface)
```

6. Attach the network interface to the stub area.

```
>> OSPF Interface 1 # ../if 2              (Select OSPF menu for IP interface 2)
>> OSPF Interface 2 # aindex 1            (Attach network to stub area index)
>> OSPF Interface 2 # enable              (Enable the stub area interface)
```

7. Apply and save the configuration changes.

```
>> OSPF Interface 2 # apply                (Global command to apply all changes)
>> OSPF Interface 2 # save                 (Global command to save all changes)
```

Example 2: Virtual Links

In the example shown in Figure 8-6, area 2 is not physically connected to the backbone as is usually required. Instead, area 2 will be connected to the backbone via a virtual link through area 1. The virtual link must be configured at each endpoint.

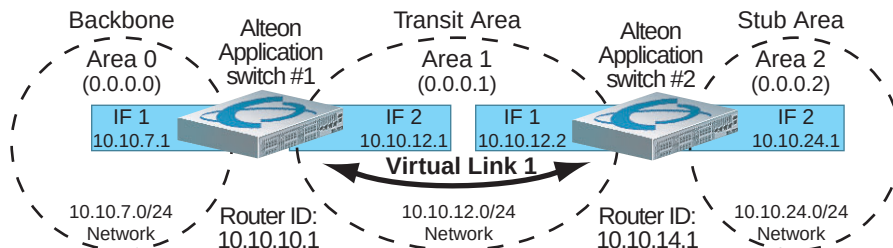


Figure 8-6 Configuring a Virtual Link

Configuring OSPF for a Virtual Link on Switch #1

1. Configure IP interfaces on each network that will be attached to the switch.

In this example, two IP interfaces are needed on Switch #1: one for the backbone network on 10.10.7.0/24 and one for the transit area network on 10.10.12.0/24.

```
>> # /cfg/ip/if 1 (Select menu for IP interface 1)
>> IP Interface 1 # addr 10.10.7.1 (Set IP address on backbone network)
>> IP Interface 1 # mask 255.255.255.0 (Set IP mask on backbone network)
>> IP Interface 1 # broad 10.10.7.255 (Set the broadcast address)
>> IP Interface 1 # enable (Enable IP interface 1)
>> IP Interface 1 # ../if 2 (Select menu for IP interface 2)
>> IP Interface 2 # addr 10.10.12.1 (Set IP address on transit area network)
>> IP Interface 2 # mask 255.255.255.0 (Set IP mask on transit area network)
>> IP Interface 2 # broad 10.10.12.255 (Set the broadcast address)
>> IP Interface 2 # enable (Enable interface 2)
```

2. Configure the router ID.

A router ID is required when configuring virtual links. Later, when configuring the other end of the virtual link on Alteon Application Switch 2, the router ID specified here will be used as the target virtual neighbor (nbr) address.

```
>> IP Interface 2 # /cfg/ip/rtrid 10.10.10.1 (Set static router ID on Alteon switch 1)
```


3. Enable OSPF.

```
>> IP # /cfg/ip/ospf/on
```

(Enable OSPF on Alteon switch 1)

4. Define the backbone.

```
>> Open Shortest Path First # aindex 0      (Select menu for area index 0)
>> OSPF Area (index) 0 # areaid 0.0.0.0    (Set the area ID for backbone area 0)
>> OSPF Area (index) 0 # type transit      (Define backbone as transit type)
>> OSPF Area (index) 0 # enable            (Enable the area)
```

5. Define the transit area.

The area that contains the virtual link must be configured as a transit area.

```
>> OSPF Area (index) 0 # ../aindex 1      (Select menu for area index 1)
>> OSPF Area (index) 1 # areaid 0.0.0.1    (Set the area ID for OSPF area 1)
>> OSPF Area (index) 1 # type transit      (Define area as transit type)
>> OSPF Area (index) 1 # enable            (Enable the area)
```

6. Attach the network interface to the backbone.

```
>> OSPF Area (index) 1 # ../if 1          (Select OSPF menu for IP interface 1)
>> OSPF Interface 1 # aindex 0            (Attach network to backbone index)
>> OSPF Interface 1 # enable              (Enable the backbone interface)
```

7. Attach the network interface to the transit area.

```
>> OSPF Interface 1 # ../if 2            (Select OSPF menu for IP interface 2)
>> OSPF Interface 2 # aindex 1            (Attach network to transit area index)
>> OSPF Interface 2 # enable              (Enable the transit area interface)
```

8. Configure the virtual link.

The nbr router ID configured in this step must be the same as the router ID that will be configured for Switch #2 in [Step 2 on page 122](#).

```
>> OSPF Interface 2 # ../virt 1           (Specify a virtual link number)
>> OSPF Virtual Link 1 # aindex 1         (Specify the transit area for the virtual link)
>> OSPF Virtual Link 1 # nbr 10.10.14.1   (Specify the router ID of the recipient)
>> OSPF Virtual Link 1 # enable           (Enable the virtual link)
```

9. Apply and save the configuration changes.

```
>> OSPF Interface 2 # apply (Global command to apply all changes)
>> OSPF Interface 2 # save (Global command to save all changes)
```

Configuring OSPF for a Virtual Link on Switch #2

1. Configure IP interfaces on each network that will be attached to OSPF areas.

Two IP interfaces are needed on Switch #2: one for the transit area network on 10.10.12.0/24 and one for the stub area network on 10.10.24.0/24.

```
>> # /cfg/ip/if 1 (Select menu for IP interface 1)
>> IP Interface 1 # addr 10.10.12.2 (Set IP address on transit area network)
>> IP Interface 1 # mask 255.255.255.0 (Set IP mask on transit area network)
>> IP Interface 1 # broad 10.10.12.255 (Set the broadcast address)
>> IP Interface 1 # enable (Enable IP interface 1)
>> IP Interface 1 # ../if 2 (Select menu for IP interface 2)
>> IP Interface 2 # addr 10.10.24.1 (Set IP address on stub area network)
>> IP Interface 2 # mask 255.255.255.0 (Set IP mask on stub area network)
>> IP Interface 2 # broad 10.10.24.255 (Set the broadcast address)
>> IP Interface 2 # enable (Enable IP interface 2)
```

2. Configure the router ID.

A router ID is required when configuring virtual links. This router ID should be the same one specified as the target virtual neighbor (nbr) on Alteon switch 1 in [Step 8 on page 121](#).

```
>> IP Interface 2 # /cfg/ip/rtrid 10.10.14.1 (Set static router ID on Alteon switch 2)
```

3. Enable OSPF.

```
>> IP# /cfg/ip/ospf/on (Enable OSPF on Alteon switch 2)
```

4. Define the backbone.

This version of Alteon OS requires that a backbone index be configured on the non-backbone end of the virtual link as follows:

```
>> Open Shortest Path First # aindex 0 (Select the menu for area index 0)
>> OSPF Area (index) 0 # areaid 0.0.0.0 (Set the area ID for OSPF area 0)
>> OSPF Area (index) 0 # enable (Enable the area)
```

5. Define the transit area.

```
>> OSPF Area (index) 0 # ../aindex 1           (Select menu for area index 1)
>> OSPF Area (index) 1 # areaid 0.0.0.1        (Set the area ID for OSPF area 1)
>> OSPF Area (index) 1 # type transit           (Define area as transit type)
>> OSPF Area (index) 1 # enable                 (Enable the area)
```

6. Define the stub area.

```
>> OSPF Area (index) 1 # ../aindex 2           (Select the menu for area index 2)
>> OSPF Area (index) 2 # areaid 0.0.0.2        (Set the area ID for OSPF area 2)
>> OSPF Area (index) 2 # type stub             (Define area as stub type)
>> OSPF Area (index) 2 # enable                 (Enable the area)
```

7. Attach the network interface to the backbone.

```
>> OSPF Area (index) 2 # ../if 1               (Select OSPF menu for IP interface 1)
>> OSPF Interface 1 # aindex 1                 (Attach network to transit area index)
>> OSPF Interface 1 # enable                   (Enable the transit area interface)
```

8. Attach the network interface to the transit area.

```
>> OSPF Interface 1 # ../if 2                 (Select OSPF menu for IP interface 2)
>> OSPF Interface 2 # aindex 2                 (Attach network to stub area index)
>> OSPF Interface 2 # enable                   (Enable the stub area interface)
```

9. Configure the virtual link.

The `nbr` router ID configured in this step must be the same as the router ID that was configured for Alteon Application Switch #1 in [Step 2 on page 120](#).

```
>> OSPF Interface 2 # ../virt 1               (Specify a virtual link number)
>> OSPF Virtual Link 1 # aindex 1             (Specify the transit area for the virtual link)
>> OSPF Virtual Link 1 # nbr 10.10.10.1        (Specify the router ID of the recipient)
>> OSPF Virtual Link 1 # enable                 (Enable the virtual link)
```

10. Apply and save the configuration changes.

```
>> OSPF Interface 2 # apply                   (Global command to apply all changes)
>> OSPF Interface 2 # save                     (Global command to save all changes)
```

Other Virtual Link Options

- You can use redundant paths by configuring multiple virtual links.
- Only the endpoints of the virtual link are configured. The virtual link path may traverse multiple routers in an area as long as there is a routable path between the endpoints.

Example 3: Summarizing Routes

By default, ABRs advertise all the network addresses from one area into another area. Route summarization can be used for consolidating advertised addresses and reducing the perceived complexity of the network.

If the network IP addresses in an area are assigned to a contiguous subnet range, you can configure the ABR to advertise a single summary route that includes all the individual IP addresses within the area.

The following example shows one summary route from area 1 (stub area) injected into area 0 (the backbone). The summary route consists of all IP addresses from 36.128.192.0 through 36.128.254.255 except for the routes in the range 36.128.200.0 through 36.128.200.255.

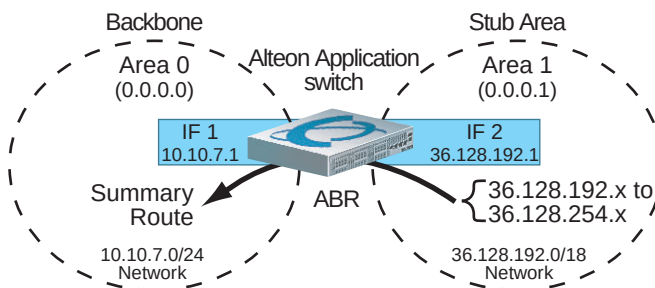


Figure 8-7 Summarizing Routes

NOTE – You can specify a range of addresses to *prevent* advertising by using the `hide` option. In this example, routes in the range 36.128.200.0 through 36.128.200.255 are kept private.

Follow this procedure to configure OSPF support as shown in [Figure 8-7](#):

1. Configure IP interfaces for each network which will be attached to OSPF areas.

```
>> # /cfg/ip/if 1 (Select menu for IP interface 1)
>> IP Interface 1 # addr 10.10.7.1 (Set IP address on backbone network)
>> IP Interface 1 # mask 255.255.255.0 (Set IP mask on backbone network)
>> IP Interface 1 # broad 10.10.7.255 (Set the broadcast address)
>> IP Interface 1 # ena (Enable IP interface 1)
>> IP Interface 1 # ../if 2 (Select menu for IP interface 2)
>> IP Interface 2 # addr 36.128.192.1 (Set IP address on stub area network)
>> IP Interface 2 # mask 255.255.192.0 (Set IP mask on stub area network)
>> IP Interface 2 # broad 36.128.255.255 (Set the broadcast address)
>> IP Interface 2 # ena (Enable IP interface 2)
```

2. Enable OSPF.

```
>> IP Interface 2 # /cfg/ip/ospf/on (Enable OSPF on the Alteon switch)
```

3. Define the backbone.

```
>> Open Shortest Path First # aindex 0 (Select menu for area index 0)
>> OSPF Area (index) 0 # areaid 0.0.0.0 (Set the ID for backbone area 0)
>> OSPF Area (index) 0 # type transit (Define backbone as transit type)
>> OSPF Area (index) 0 # enable (Enable the area)
```

4. Define the stub area.

```
>> OSPF Area (index) 0 # ../aindex 1 (Select menu for area index 1)
>> OSPF Area (index) 1 # areaid 0.0.0.1 (Set the area ID for OSPF area 1)
>> OSPF Area (index) 1 # type stub (Define area as stub type)
>> OSPF Area (index) 1 # enable (Enable the area)
```

5. Attach the network interface to the backbone.

```
>> OSPF Area (index) 1 # ../if 1 (Select OSPF menu for IP interface 1)
>> OSPF Interface 1 # aindex 0 (Attach network to backbone index)
>> OSPF Interface 1 # enable (Enable the backbone interface)
```

6. Attach the network interface to the stub area.

```
>> OSPF Interface 1 # ../if 2 (Select OSPF menu for IP interface 2)
>> OSPF Interface 2 # aindex 1 (Attach network to stub area index)
>> OSPF Interface 2 # enable (Enable the stub area interface)
```

7. Configure route summarization by specifying the starting address and mask of the range of addresses to be summarized.

>> OSPF Interface 2 # ../range 1	<i>(Select menu for summary range)</i>
>> OSPF Summary Range 1 # addr 36.128.192.0	<i>(Set base IP address of summary range)</i>
>> OSPF Summary Range 1 # mask 255.255.192.0	<i>(Set mask address for summary range)</i>
>> OSPF Summary Range 1 # aindex 0	<i>(Inject summary route into backbone)</i>
>> OSPF Summary Range 1 # enable	<i>(Enable summary range)</i>

8. Use the hide command to prevent a range of addresses from advertising to the backbone.

>> OSPF Interface 2 # ../range 2	<i>(Select menu for summary range)</i>
>> OSPF Summary Range 2 # addr 36.128.200.0	<i>(Set base IP address)</i>
>> OSPF Summary Range 2 # mask 255.255.255.0	<i>(Set mask address)</i>
>> OSPF Summary Range 2 # hide enable	<i>(Hide the range of addresses)</i>

9. Apply and save the configuration changes.

>> OSPF Summary Range 2 # apply	<i>(Global command to apply all changes)</i>
>> OSPF Summary Range 2 # save	<i>(Global command to save all changes)</i>

Example 4: Host Routes

The Alteon OS implementation of OSPF includes host routes. Host routes are used for advertising network device IP addresses to external networks and allows for Server Load Balancing (SLB) within OSPF. It also makes ABR load sharing and failover possible.

Consider the example network in Figure 8-8. Both Alteon Application Switches have access to servers with identical content and are configured with the same virtual server IP addresses: 10.10.10.1 and 10.10.10.2. Alteon Application Switch #1 is given a host route with a low cost for virtual server 10.10.10.1 and another host route with a high cost for virtual server 10.10.10.2. Alteon Application Switch #2 is configured with the same hosts but with the costs reversed; one host route has a high cost for virtual server 10.10.10.1 and another has a low cost for virtual server 10.10.10.2.

All four host routes are injected into the upstream router and advertised externally. Traffic comes in for both virtual server IP addresses (10.10.10.1 and 10.10.10.2). The upstream router sees that both addresses exist on both Alteon switches and uses the host route with the lowest cost for each. Traffic for 10.10.10.1 goes to Alteon switch #1 because its host route has the lowest cost for that address. Traffic for 10.10.10.2 goes to Alteon switch #2 because its host route has the lowest cost. This effectively shares the load among ABRs. Both Alteon switches then use standard server load balancing to distribute traffic among available real servers.

In addition, if one of the Alteon switches were to fail, the upstream routing device would forward the traffic to the ABR whose host route has the next lowest cost. In this example, the remaining Alteon switch would assume the entire load for both virtual servers.

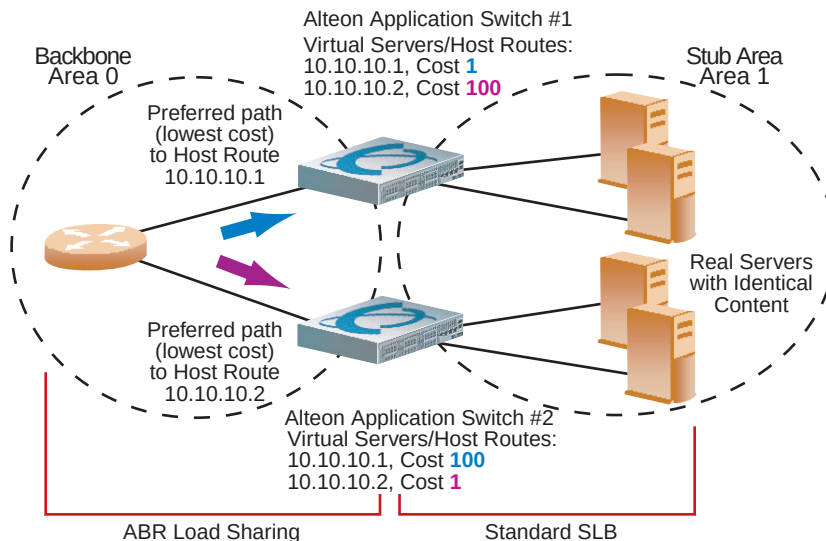


Figure 8-8 Configuring OSPF Host Routes

Configuring OSPF for Host Routes on Alteon Application Switch #1

1. Configure IP interfaces for each network that will be attached to OSPF areas.

>> Virtual server 1 # /cfg/ip/if 1	(Select menu for IP interface 1)
>> IP Interface 1 # addr 10.10.10.5	(Set IP address on backbone network)
>> IP Interface 1 # enable	(Enable IP interface 1)
>> IP Interface 1 # ../if 2	(Select menu for IP interface 2)
>> IP Interface 2 # addr 100.100.100.40	(Set IP address on stub area network)
>> IP Interface 2 # enable	(Enable IP interface 2)

2. Configure basic SLB parameters.

Alteon Application Switch 1 is connected to two real servers. Each real server is given an IP address and is placed in the same real server group.

>> # /cfg/slb/real 1	(Select menu for real server 1)
>> Real server 1 # rip 100.100.100.25	(Set the IP address for real server 1)
>> Real server 1 # ena	(Enable the real server)
>> Real server 1 # ../real 2	(Select menu for real server 2)
>> Real server 2 # rip 100.100.100.26	(Set the IP address for real server 2)
>> Real server 2 # ena	(Enable the real server)
>> Real server 2 # ../group 1	(Select menu for real server group 1)
>> Real server group 1 # add 1	(Add real server 1 to group)
>> Real server group 1 # add 2	(Add real server 2 to group)
>> Real server group 1 # enable	(Enable the group)
>> Real server group 1 # ../on	(Turn SLB on)

3. Configure client and server processing on specific ports.

>> Layer 4# port 4	(Select switch port 4)
>> SLB Port 4 # client ena	(Enable client processing on Port 4)
>> SLB Port 4 # ../port 5	(Select switch Port 5)
>> SLB Port 5 # server ena	(Enable server processing on Port 5)

4. Enable direct access mode.

>> Layer 4 Port 5# ../adv	(Select the SLB advance menu)
>> Layer 4 Advanced# direct ena	(Enable DAM)
>> Layer 4 Advanced# ..	(Return to the SLB menu)

5. Configure the primary virtual server.

Alteon Application Switch # 1 will be preferred for virtual server 10.10.10.1.

```
>> Layer 4 # virt 1 (Select menu for virtual server 1)
>> Virtual server 1 # vip 10.10.10.1 (Set the IP address for virtual server 1)
>> Virtual server 1 # ena (Enable the virtual server)
>> Virtual server 1 # service http (Select menu for service on virtual server)
>> Virtual server 1 http service # group 1 (Use real server group 1 for http service)
```

6. Configure the backup virtual server.

Alteon Application Switch # 1 will act as a backup for virtual server 10.10.10.2. Both virtual servers in this example are configured with the same real server group and provide identical services.

```
>> Virtual server 2 http service # /cfg/slb/virt 2 (Select menu for virtual server 2)
>> Virtual server 1 # vip 10.10.10.2 (Set the IP address for virtual server 2)
>> Virtual server 1 # ena (Enable the virtual server)
>> Virtual server 1 # service http (Select menu for service on virtual server)
>> Virtual server 1 # group 1 (Use real server group 1 for http service)
```

7. Enable OSPF on Alteon Application Switch 1.

```
>> IP Interface 2 # ../ospf/on (Enable OSPF on Alteon switch 1)
```

8. Define the backbone.

```
>> Open Shortest Path First # aindex 0 (Select menu for area index 0)
>> OSPF Area (index) 0 # areaid 0.0.0.0 (Set the ID for backbone area 0)
>> OSPF Area (index) 0 # type transit (Define backbone as transit type)
>> OSPF Area (index) 0 # enable (Enable the area)
```

9. Define the stub area.

```
>> OSPF Area (index) 0 # ../aindex 1 (Select menu for area index 1)
>> OSPF Area (index) 1 # areaid 0.0.0.1 (Set the ID for stub area 1)
>> OSPF Area (index) 1 # type stub (Define area as stub type)
>> OSPF Area (index) 1 # enable (Enable the area)
```

10. Attach the network interface to the backbone.

>> OSPF Area (index) 1 # ../if 1	<i>(Select OSPF menu for IP interface 1)</i>
>> OSPF Interface 1 # aindex 0	<i>(Attach network to backbone index)</i>
>> OSPF Interface 1 # enable	<i>(Enable the backbone interface)</i>

11. Attach the network interface to the stub area.

>> OSPF Interface 1 # ../if 2	<i>(Select OSPF menu for IP interface 2)</i>
>> OSPF Interface 2 # aindex 1	<i>(Attach network to stub area index)</i>
>> OSPF Interface 2 # enable	<i>(Enable the stub area interface)</i>

12. Configure host routes.

One host route is needed for each virtual server on Alteon Application Switch 1. Since virtual server 10.10.10.1 is preferred for Alteon Application Switch 1, its host route has a low cost. Because virtual server 10.10.10.2 is used as a backup in case Alteon Application Switch 2 fails, its host route has a high cost.

NOTE – You do not need to enable redistribution (`cfg/ip/ospf/redist`) if you configure virtual server routes as host routes.

>> OSPF Interface 2 # ../host 1	<i>(Select menu for host route 1)</i>
>> OSPF Host Entry 1 # addr 10.10.10.1	<i>(Set IP address same as virtual server 1)</i>
>> OSPF Host Entry 1 # aindex 0	<i>(Inject host route into backbone area)</i>
>> OSPF Host Entry 1 # cost 1	<i>(Set low cost for preferred path)</i>
>> OSPF Host Entry 1 # enable	<i>(Enable the host route)</i>
>> OSPF Host Entry 1 # ../host 2	<i>(Select menu for host route 2)</i>
>> OSPF Host Entry 2 # addr 10.10.10.2	<i>(Set IP address same as virtual server 2)</i>
>> OSPF Host Entry 2 # aindex 0	<i>(Inject host route into backbone area)</i>
>> OSPF Host Entry 2 # cost 100	<i>(Set high cost for use as backup path)</i>
>> OSPF Host Entry 2 # enable	<i>(Enable the host route)</i>

NOTE – When a service goes down, the corresponding host route is removed from advertising.

13. Apply and save the configuration changes.

>> OSPF Host Entry 2 # apply	<i>(Global command to apply all changes)</i>
>> OSPF Host Entry 2 # save	<i>(Global command to save all changes)</i>

Configuring OSPF for Host Routes on Alteon Application Switch 2

1. Configure basic SLB parameters.

Alteon Application Switch 2 is connected to two real servers. Each real server is given an IP address and is placed in the same real server group.

```
>> # /cfg/slb/real 1                (Select menu for real server 1)
>> Real server 1 # rip 100.100.100.27 (Set the IP address for real server 1)
>> Real server 1 # enable            (Enable the real server)
>> Real server 1 # ../real 2        (Select menu for real server 2)
>> Real server 2 # rip 100.100.100.28 (Set the IP address for real server 2)
>> Real server 2 # enable            (Enable the real server)
>> Real server 2 # ../group 1       (Select menu for real server group 1)
>> Real server group 1 # add 1       (Add real server 1 to group)
>> Real server group 1 # add 2       (Add real server 2 to group)
>> Real server group 1 # enable      (Enable the group)
>> Real server group 1 # ../on       (Turn SLB on)
```

2. Configure the virtual server parameters.

The same virtual servers are configured as on Alteon Application Switch 1.

```
>> Layer 4 # virt 1                (Select menu for virtual server 1)
>> Virtual server 1 # vip 10.10.10.1 (Set the IP address for virtual server 1)
>> Virtual server 1 # enable        (Enable the virtual server)
>> Virtual server 1 # service http  (Select menu for service on virtual server)
>> Virtual server 1 http service # group 1 (Use real server group 1 for http service)
>> Virtual server 2 http service # /cfg/slb/virt 2 (Select menu for virtual server 2)
>> Virtual server 1 # vip 10.10.10.2 (Set the IP address for virtual server 2)
>> Virtual server 1 # enable        (Enable the virtual server)
>> Virtual server 1 # service http  (Select menu for service on virtual server)
>> Virtual server 1 # group 1       (Use real server group 1 for http service)
```

3. Configure IP interfaces for each network that will be attached to OSPF areas.

```
>> Virtual server 1 # /cfg/ip/if 1    (Select menu for IP Interface 1)
>> IP Interface 1 # addr 10.10.10.6   (Set IP address on backbone network)
>> IP Interface 1 # enable            (Enable IP interface 1)
>> IP Interface 1 # ../if 2          (Select menu for IP Interface 2)
>> IP Interface 2 # addr 100.100.100.41 (Set IP address on stub area network)
>> IP Interface 2 # enable            (Enable IP interface 2)
```

4. Enable OSPF on Alteon Application Switch #2.

>> IP Interface 2 # ../ospf/on	<i>(Enable OSPF on Alteon switch #2)</i>

5. Define the backbone.

>> Open Shortest Path First # aindex 0	<i>(Select menu for area index 0)</i>
>> OSPF Area (index) 0 # areaid 0.0.0.0	<i>(Set the ID for backbone area 0)</i>
>> OSPF Area (index) 0 # type transit	<i>(Define backbone as transit type)</i>
>> OSPF Area (index) 0 # enable	<i>(Enable the area)</i>

6. Define the stub area.

>> OSPF Area (index) 0 # ../aindex 1	<i>(Select menu for area index 1)</i>
>> OSPF Area (index) 1 # areaid 0.0.0.1	<i>(Set the ID for stub area 1)</i>
>> OSPF Area (index) 1 # type stub	<i>(Define area as stub type)</i>
>> OSPF Area (index) 1 # enable	<i>(Enable the area)</i>

7. Attach the network interface to the backbone.

>> OSPF Area (index) 1 # ../if 1	<i>(Select OSPF menu for IP interface 1)</i>
>> OSPF Interface 1 # aindex 0	<i>(Attach network to backbone index)</i>
>> OSPF Interface 1 # enable	<i>(Enable the backbone interface)</i>

8. Attach the network interface to the stub area.

>> OSPF Interface 1 # ../if 2	<i>(Select OSPF menu for IP interface 2)</i>
>> OSPF Interface 2 # aindex 1	<i>(Attach network to stub area index)</i>
>> OSPF Interface 2 # enable	<i>(Enable the stub area interface)</i>

9. Configure host routes.

Host routes are configured just like those on Alteon Application Switch 1, except their costs are *reversed*. Since virtual server 10.10.10.2 is preferred for Alteon Application Switch 2, its host route has been given a low cost. Because virtual server 10.10.10.1 is used as a backup in case Alteon switch 1 fails, its host route has been given a high cost.

>> OSPF Interface 2 # ../host 1	<i>(Select menu for host route 1)</i>
>> OSPF Host Entry 1 # addr 10.10.10.1	<i>(Set IP address same as virtual server 1)</i>
>> OSPF Host Entry 1 # aindex 0	<i>(Inject host route into backbone area)</i>
>> OSPF Host Entry 1 # cost 100	<i>(Set high cost for use as backup path)</i>
>> OSPF Host Entry 1 # enable	<i>(Enable the host route)</i>
>> OSPF Host Entry 1 # ../host 2	<i>(Select menu for host route 2)</i>
>> OSPF Host Entry 2 # addr 10.10.10.2	<i>(Set IP address same as virtual server 2)</i>
>> OSPF Host Entry 2 # aindex 0	<i>(Inject host route into backbone area)</i>
>> OSPF Host Entry 2 # cost 1	<i>(Set low cost for primary path)</i>
>> OSPF Host Entry 2 # enable	<i>(Enable the host route)</i>

10. Apply and save the configuration changes.

>> OSPF Host Entry 2 # apply	<i>(Global command to apply all changes)</i>
>> OSPF Host Entry 2 # save	<i>(Global command to save all changes)</i>

Verifying OSPF Configuration

Use the following commands to verify the OSPF configuration on your switch:

- `/info/ospf/general`
- `/info/ospf/nbr`
- `/info/ospf/dbase/dbsum`
- `/info/ospf/route`
- `/stats/route`

Refer to the *Alteon OS Command Reference* for information on the above commands.

Part 3: Application Switching Fundamentals

Internet traffic consists of myriad services and applications which use the Internet Protocol (IP) for data delivery. IP, however, is not optimized for all the various applications. Application switching goes beyond IP and makes intelligent switching decisions based on the application and its data. This sections details the following fundamental switching features:

- Server Load Balancing
- Filtering
- Application Redirection
- Health Checking
- High Availability



CHAPTER 9

Server Load Balancing

Server Load Balancing (SLB) allows you to configure the Alteon Application Switch to balance user session traffic among a pool of available servers that provide shared services. The following sections in this chapter describe how to configure and use SLB:

- [“Understanding Server Load Balancing” on page 138](#). This section discusses the benefits of server load balancing and how it works.
- [“Implementing Basic Server Load Balancing” on page 141](#). This section discusses how implementing SLB provides reliability, performance, and ease of maintenance on your network.
 - [“Network Topology Requirements” on page 142](#). This section provides key requirements to consider before deploying server load balancing.
 - [“Configuring Server Load Balancing” on page 144](#).
 - [“Additional Server Load Balancing Options” on page 148](#).
- [“Extending SLB Topologies” on page 156](#). This section discusses proxy IP addresses, mapping a virtual port to real ports, monitoring real server ports, and delayed binding.
- [“Load Balancing Special Services” on page 170](#). This section discusses load balancing based on special services, such as source IP addresses, FTP, RTSP, DNS, WAP, IDS, and WAN link.

For additional information on SLB commands, see your *Alteon OS Command Reference*.

Understanding Server Load Balancing

SLB benefits your network in a number of ways:

- Increased efficiency for server utilization and network bandwidth
With SLB, your Alteon Application Switch is aware of the shared services provided by your server pool and can then balance user session traffic among the available servers. Important session traffic gets through more easily, reducing user competition for connections on overutilized servers. For even greater control, traffic is distributed according to a variety of user-selectable rules.
- Increased reliability of services to users
If any server in a server pool fails, the remaining servers continue to provide access to vital applications and data. The failed server can be brought back up without interrupting access to services.
- Increased scalability of services
As users are added and the server pool's capabilities are saturated, new servers can be added to the pool transparently.

Identifying Your Network Needs

SLB may be the right option for addressing these vital network concerns:

- A single server no longer meets the demand for its particular application.
- The connection from your LAN to your server overloads the server's capacity.
- Your NT and UNIX servers hold critical application data and must remain available even in the event of a server failure.
- Your Web site is being used as a way to do business and for taking orders from customers. It must not become overloaded or unavailable.
- You want to use multiple servers or hot-standby servers for maximum server uptime.
- You must be able to scale your applications to meet client and LAN request capacity.
- You can't afford to continue using an inferior load-balancing technique, such as DNS round robin or a software-only system.

How Server Load Balancing Works

In an average network that employs multiple servers without server load balancing, each server usually specializes in providing one or two unique services. If one of these servers provides access to applications or data that is in high demand, it can become overutilized. Placing this kind of strain on a server can decrease the performance of the entire network as user requests are rejected by the server and then resubmitted by the user stations. Ironically, over-utilization of key servers often happens in networks where other servers are actually available.

The solution to getting the most from your servers is SLB. With this software feature, the switch is aware of the services provided by each server. The switch can direct user session traffic to an appropriate server, based on a variety of load-balancing algorithms.

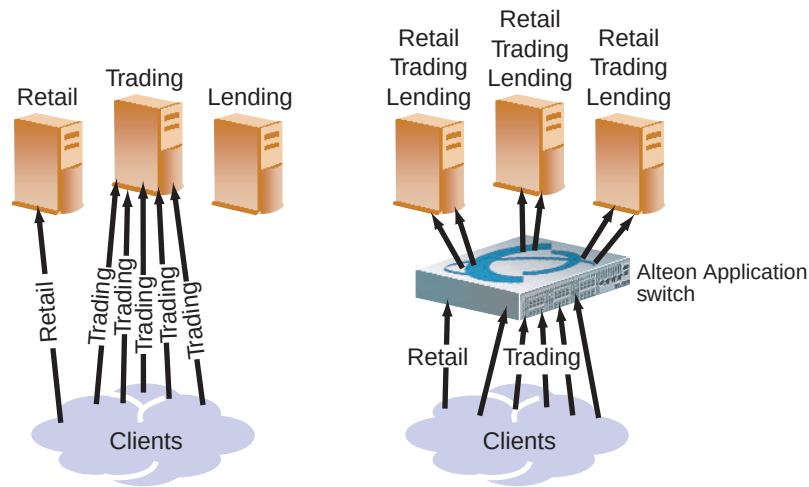


Figure 9-1 Traditional Versus SLB Network Configurations

To provide load balancing for any particular type of service, each server in the pool must have access to identical content, either directly (duplicated on each server) or through a back-end network (mounting the same file system or database server).

The Alteon Application Switch, with SLB software, acts as a front-end to the servers, interpreting user session requests and distributing them among the available servers. Load balancing in Alteon OS can be done in the following ways:

- Virtual server-based load balancing

This is the traditional load balancing method. The switch is configured to act as a virtual server and is given a virtual server IP address (or range of addresses) for each collection of services it will distribute. Depending on your switch model, there can be as many as 256 virtual servers on the switch, each distributing up to eight different services (up to a total of 2048 services).

Each virtual server is assigned a list of the IP addresses (or range of addresses) of the real servers in the pool where its services reside. When the user stations request connections to a service, they will communicate with a virtual server on the switch. When the switch receives the request, it binds the session to the IP address of the best available real server and remaps the fields in each frame from virtual addresses to real addresses.

HTTP, IP, FTP, RTSP, IDS, and static session WAP are examples of some of the services that use virtual servers for load balancing.

- Filtered-based load balancing

A filter allows you to control the types of traffic permitted through the switch. Filters are configured to allow, deny, or redirect traffic according to the IP address, protocol, or Layer 4 port criteria. In filtered-based load balancing, a filter is used to redirect traffic to a real server group. If the group is configured with more than one real server entry, redirected traffic is load balanced among the available real servers in the group.

Firewalls, WAP with RADIUS snooping, IDS, and WAN links use redirection filters to load balance traffic.

- Content-based load balancing

Content-based load balancing uses Layer 7 application data (such as URL, cookies, and Host Headers) to make intelligent load balancing decisions.

URL-based load balancing, browser-smart load balancing, and cookie-based preferential load balancing are a few examples of content-based load balancing.

Implementing Basic Server Load Balancing

Consider a situation where customer Web sites are being hosted by a popular Web hosting company and/or Internet Service Provider (ISP). The Web content is relatively static and is kept on a single NFS server for easy administration. As the customer base increases, the number of simultaneous Web connection requests also increases.

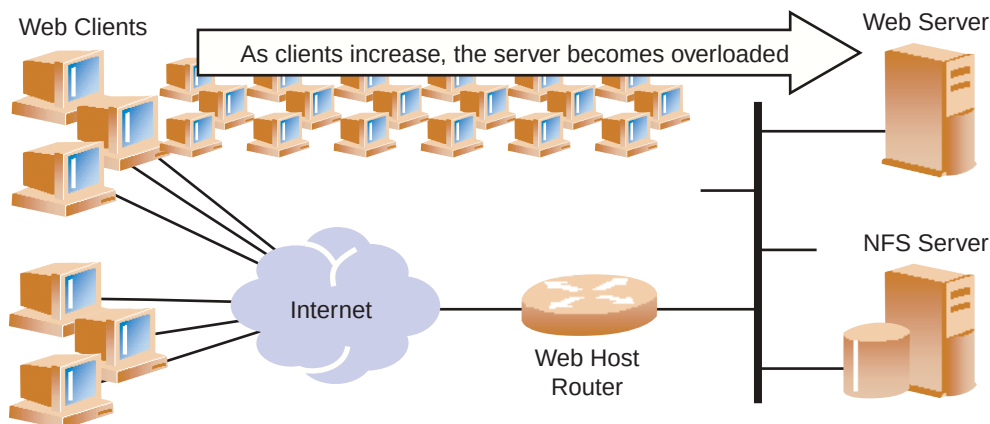


Figure 9-2 Web Hosting Configuration Without SLB

Such a company has three primary needs:

- Increased server availability
- Server performance scalable to match new customer demands
- Easy administration of network and servers

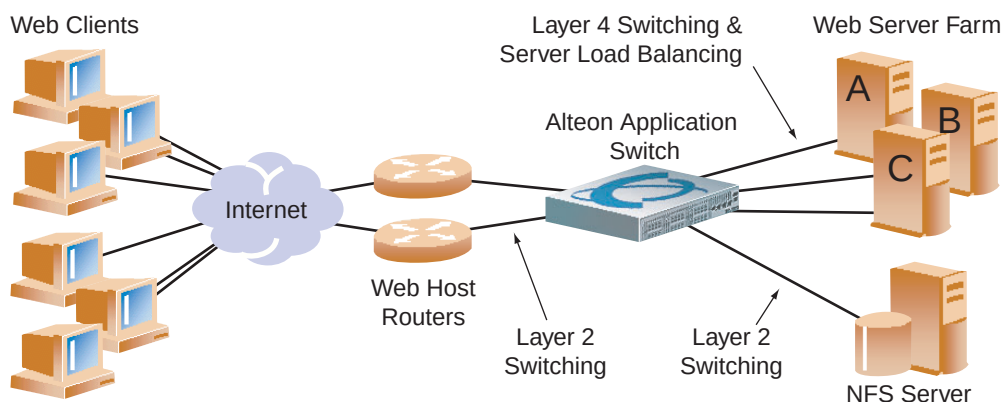


Figure 9-3 Web Hosting with SLB Solutions

All the issues can be addressed by adding an Alteon Application Switch with SLB software.

- Reliability is increased by providing multiple paths from the clients to the Alteon Application Switch and by accessing a pool of servers with identical content. If one server fails, the others can take up the additional load.
- Performance is improved by balancing the Web request load across multiple servers. More servers can be added at any time to increase processing power.
- For ease of maintenance, servers can be added or removed dynamically, without interrupting shared services.

Network Topology Requirements

When deploying SLB, there are a few key aspects to consider:

- In standard SLB, all client requests to a virtual server IP address and all responses from the real servers must pass through the switch, as shown in [Figure 9-4](#). If there is a path between the client and the real servers that does not pass through the switch, the Alteon Application Switch can be configured to proxy requests in order to guarantee that responses use the correct path (see “[Proxy IP Addresses](#)” on page 156).

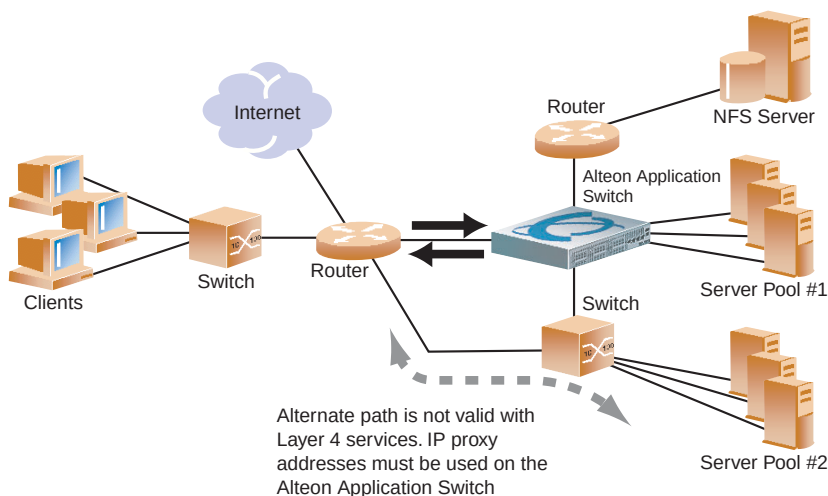


Figure 9-4 SLB Client/Server Traffic Routing

- Identical content must be available to each server in the same pool. Either of these methods can be used:
 - Static applications and data are duplicated on each real server in the pool.
 - Each real server in the pool has access to the same data through use of a shared file system or back-end database server.

- Some services require that a series of client requests go to the same real server so that session-specific state data can be retained between connections. Services of this nature include Web search results, multi-page forms that the user fills in, or custom Web-based applications typically created using `cgi-bin` scripts. Connections for these types of services must be configured as *persistent* (see [Chapter 18, “Persistence”](#)) or must use the *minmisses* or *hash metrics* (see [“Metrics for Real Server Groups” on page 151](#)).
- Clients and servers can be connected through the same switch port. Each port in use on the switch can be configured to process client requests, server traffic, or both. You can enable or disable processing on a port independently for each type of Layer 4 traffic.
 - Layer 4 client processing: Ports that are configured to process client request traffic provide address translation from the virtual server IP to the real server IP address.
 - Layer 4 server processing: Ports that are configured to process server responses to client requests provide address translation from the real server IP address to the virtual server IP address. These ports require real servers to be connected to the Alteon Application Switch directly or through a hub, router, or another switch.

NOTE – Switch ports configured for Layer 4 client/server processing can simultaneously provide Layer 2 switching and IP routing functions.

Consider the following network topology:

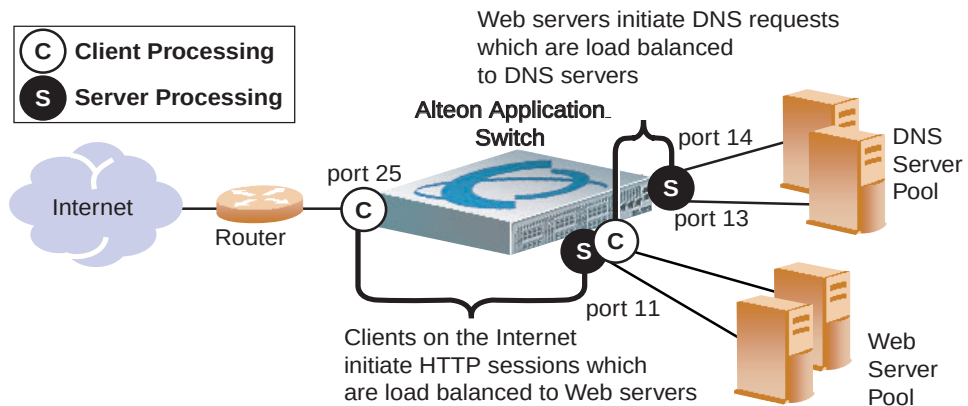


Figure 9-5 Example Network for Client/Server Port Configuration

In [Figure 9-5](#), the switch load balances traffic to a Web server pool and to a Domain Name System (DNS) server pool. The switch port connected to the Web server pool (port 11) is asked to perform both server and client processing.

Some topologies require special configuration. For example, if clients were added to Switch B as shown in [Figure 9-5](#), these clients could not access the Web server pool using SLB services, except through a proxy IP address that is configured on port 2 of the Alteon Application Switch.

Configuring Server Load Balancing

This section describes the steps for configuring an SLB Web hosting solution. In the following procedure, many of the SLB options are left to their default values. See [“Additional Server Load Balancing Options” on page 148](#) for more options. Before you start configuring, you must be connected to the switch CLI as the administrator.

NOTE – For details about any of the menu commands described in this example, see your *Alteon OS Command Reference*.

1. Assign an IP address to each of the real servers in the server pool.

The real servers in any given real server group must have an IP route to the switch that performs the SLB functions. This IP routing is most easily accomplished by placing the switches and servers on the same IP subnet, although advanced routing techniques can be used as long as they do not violate the topology rules outlined in [“Network Topology Requirements” on page 142](#).

For this example, the three hosts (real servers) have been given the following IP addresses on the same IP subnet:

Table 9-1 Web Host Example: Real Server IP Addresses

Real Server	IP address
Server A	200.200.200.2
Server B	200.200.200.3
Server C	200.200.200.4

NOTE – An `imask` option can be used to define a range of IP addresses for real and virtual servers (see [“IP Address Ranges Using imask” on page 149](#)).

2. Define an IP interface on the switch.

The switch must have an IP route to all of the real servers that receive switching services. For SLB, the switch uses this path to determine the level of TCP/IP reach of the real servers.

To configure an IP interface for this example, enter these commands from the CLI:

```
>> # /cfg/ip/if 1                (Select IP interface 1)
>> IP Interface 1# addr 200.200.200.100 (Assign IP address for the interface)
>> IP Interface 1# ena            (Enable IP interface 1)
```

NOTE – The IP interface and the real servers must belong to the same VLAN, if they are in the same subnet. This example assumes that all ports and IP interfaces use default VLAN 1, requiring no special VLAN configuration for the ports or IP interface.

3. Define each real server.

For each real server, you must assign a real server number, specify its actual IP address, and enable the real server. For example:

```
>> IP Interface 1# /cfg/slb/real 1    (Server A is real server 1)
>> Real server 1# rip 200.200.200.2  (Assign Server A IP address)
>> Real server 1# ena                (Enable real server 1)
>> Real server 1# ../real 2          (Server B is real server 2)
>> Real server 2# rip 200.200.200.3  (Assign Server B IP address)
>> Real server 2# ena                (Enable real server 2)
>> Real server 2# ../real 3          (Server C is real server 3)
>> Real server 3# rip 200.200.200.4  (Assign Server C IP address)
>> Real server 3# ena                (Enable real server 3)
```

4. Define a real server group and add the three real servers to the service group.

```
>> Real server 3# /cfg/slb/group 1    (Select real server group 1)
>> Real server group 1# add 1         (Add real server 1 to group 1)
>> Real server group 1# add 2         (Add real server 2 to group 1)
>> Real server group 1# add 3         (Add real server 3 to group 1)
```

5. Define a virtual server.

All client requests will be addressed to a virtual server IP address on a virtual server defined on the switch. Clients acquire the virtual server IP address through normal DNS resolution. In this example, HTTP is configured as the only service running on this virtual server, and this service is associated with the real server group. For example:

```
>> Real server group 1# /cfg/slb/virt 1      (Select virtual server 1)
>> Virtual server 1# vip 200.200.200.1      (Assign a virtual server IP address)
>> Virtual server 1# ena                    (Enable the virtual server)
>> Virtual server 1# service http           (Select the HTTP service menu)
>> Virtual server 1 http Service# group 1   (Associate virtual port to real group)
```

NOTE – This configuration is not limited to HTTP Web service. Other TCP/IP services can be configured in a similar fashion. For a list of other well-known services and ports, see “Well-Known Application Ports” on page 148. To configure multiple services, see “Configuring Multiple Services” on page 150.

6. Define the port settings.

In this example, the following ports are being used on the Alteon Application Switch:

Table 9-2 Web Host Example: Port Usage

Port	Host	L4 Processing
1	Server A serves SLB requests.	Server
2	Server B serves SLB requests.	Server
3	Server C serves SLB requests.	Server
4	Back-end NFS server provides centralized content for all three real servers. This port does not require switching features.	None
5	Client router A connects the switch to the Internet where client requests originate.	Client
6	Client router B connects the switch to the Internet where client requests originate.	Client

The ports are configured as follows:

>> Virtual server 1# /cfg/slb/port 1	(Select physical switch port 1)
>> SLB port 1# server ena	(Enable server processing on port 1)
>> SLB port 1# ../port 2	(Select physical switch port 2)
>> SLB port 2# server ena	(Enable server processing on port 2)
>> SLB port 2# ../port 3	(Select physical switch port 3)
>> SLB port 3# server ena	(Enable server processing on port 3)
>> SLB port 3# ../port 5	(Select physical switch port 5)
>> SLB port 5# client ena	(Enable client processing on port 5)
>> SLB port 5# ../port 6	(Select physical switch port 6)
>> SLB port 6# client ena	(Enable client processing on port 6)

7. Enable, apply, and verify the configuration.

>> SLB port 6# /cfg/slb	(Select the SLB Menu)
>> Layer 4# on	(Turn Server Load Balancing on)
>> Layer 4# apply	(Make your changes active)
>> Layer 4# cur	(View current settings)

Examine the resulting information. If any settings are incorrect, make the appropriate changes.

8. Save your new configuration changes.

>> Layer 4# save	(Save for restore after reboot)

NOTE – You must apply any changes in order for them to take effect, and you must save changes if you wish them to remain in effect after switch reboot.

9. Check the SLB information.

>> Layer 4# /info/slb/dump	(View SLB information)

Check that all SLB parameters are working according to expectation. If necessary, make any appropriate configuration changes and then check the information again.

Additional Server Load Balancing Options

In the previous section (“[Configuring Server Load Balancing](#)” on page 144), many of the SLB options are left to their default values. The following configuration options can be used to customize SLB on your Alteon Application Switch:

- “Supported Services and Applications” on page 148
- “Disabling and Enabling Real Servers” on page 149
- “IP Address Ranges Using imask” on page 149
- “Health Checks for Real Servers” on page 150
- “Configuring Multiple Services” on page 150
- “Metrics for Real Server Groups” on page 151
- “Weights for Real Servers” on page 154
- “Connection Time-outs for Real Servers” on page 154
- “Maximum Connections for Real Servers” on page 155
- “Backup/Overflow Servers” on page 155

Supported Services and Applications

Each virtual server can be configured to support up to eight services, limited to a total of 256 services per switch. Using the `/cfg/slb/virt <virtual server number>/service` option, the following TCP/UDP applications can be specified:

NOTE – The service number specified on the switch must match the service specified on the server.

Table 9-3 Well-Known Application Ports

Number	TCP/UDP Application	Number	TCP/UDP Application	Number	TCP/UDP Application
20	ftp-data	79	finger	179	bgp
21	ftp	80	http	194	irc
22	ssh	109	pop2	220	imap3
23	telnet	110	pop3	389	ldap
25	smtp	111	sunrpc	443	https
37	time	119	nntp	520	rip
42	name	123	ntp	554	rtsp
43	whois	143	imap	1645, 1812	Radius
53	domain	144	news	1813	Radius Accounting
69	tftp	161	snmp	1985	hsrp
70	gopher	162	snmptrap		

NOTE – Load balancing some applications (such as FTP and RTSP) require special configuration. See “[Load Balancing Special Services](#)” on page 170 for more information.

Disabling and Enabling Real Servers

If you need to reboot a server, make sure that new sessions are not sent to the real server and that old sessions are not discarded. When the session count reaches zero, you may shut down the server. Use the following command to disable real servers:

```
>> # /oper/slb/dis <real server number>
```

When maintenance is complete, use the following command to enable the real server:

```
>> # /oper/slb/ena <real server number>
```

IP Address Ranges Using imask

The `imask` option lets you define a range of IP addresses for the real and virtual servers configured under SLB. By default, the `imask` setting is 255.255.255.255, which means that each real and virtual server represents a single IP address. An `imask` setting of 255.255.255.0 would mean that each real and virtual server represents 256 IP addresses. Consider the following example:

- A virtual server is configured with an IP address of 172.16.10.1.
- Real servers 172.16.20.1 and 172.16.30.1 are assigned to service the virtual server.
- The `imask` is set to 255.255.255.0.

If the client request was sent to virtual server IP address 172.16.10.45, the unmasked portion of the address (0.0.0.45) gets mapped directly to whichever real server IP address is selected by the SLB algorithm. Thus, the request would be sent to either 172.16.20.45 or 172.16.30.45.

Health Checks for Real Servers

Determining health for each real server is a necessary function for SLB. By default for TCP services, the switch checks health by opening a TCP connection to each service port configured as part of each service. For more information, see [“Configuring Multiple Services” on page 150](#). For UDP services, the switch pings servers to determine their status.

The switch checks each service on each real server every two seconds. If the real server is busy processing connections, it may not respond to a health check. By default, if a service does not respond to four consecutive health checks, the switch declares the service unavailable. As shown below, the health check interval and the number of retries can be changed:

>> # /cfg/slb/real <real server number>	(Select the real server)
>> Real server# inter 4	(Check real server every 4 seconds)
>> Real server# retry 6	(Declare down after 6 checks fail)

For more complex health-checking strategies, see [Chapter 12, “Health Checking.”](#)

Configuring Multiple Services

When you configure multiple services in the same group, their health checks will be dependent on each other. If a real server fails a health check for a service, then the status of the real server for the second service appears as “blocked.”

Independent Services. If you are configuring two independent services such as FTP and SMTP—where the real server failure on one service does not affect other services that the real server supports, then configure two groups with the same real servers, but with different services. If a real server configured for both FTP and SMTP fails FTP, the real server is still available for SMTP. This allows the services to act independently even though they are using the same real servers.

Dependent Services. If you are configuring two dependent services such as HTTP and HTTPS—where the real server failure on one service blocks the real server for other services, then configure a single group with multiple services. If a real server configured for both HTTP and HTTPS fails for the HTTP service, then the server is blocked from supporting any HTTPS requests. The switch will block HTTPS requests, (even though HTTPS has not failed) until the HTTP service becomes available again. This helps in troubleshooting so you know which service has failed.

Metrics for Real Server Groups

Metrics are used for selecting which real server in a group will receive the next client connection. The available metrics `minmisses` (minimum misses), `hash`, `leastconns` (least connections), `roundrobin`, `bandwidth`, and `response` (response time) are explained in detail below. The default metric is `leastconns`.

To change a real server group metric to `minmisses`, for example, enter:

<code>>> # /cfg/slb/group <group number></code>	<i>(Select the real server group)</i>
<code>>> Real server group# metric minmisses</code>	<i>(Use minmisses metric)</i>

Minimum Misses

The `minmisses` metric is optimized for application redirection. It uses IP address information in the client request to select a server. When selecting a server, the switch calculates a value for each available real server based on the relevant IP address information. The server with highest value is assigned the connection. This metric attempts to minimize the disruption of persistency when servers are removed from service. This metric should be used only when persistence is a must.

The specific IP address information used depends on the application:

- For application redirection, the client destination IP address is used. All requests for a specific IP destination address is sent to the same server. This metric is particularly useful in caching applications, helping to maximize successful cache hits. Best statistical load balancing is achieved when the IP address destinations of load-balanced frames are spread across a broad range of IP subnets.
- For SLB, the client source IP address and real server IP address are used. All requests from a specific client are sent to the same server. This metric is useful for applications where client information must be retained on the server between sessions. With this metric, server load becomes most evenly balanced as the number of active clients with different source or destination addresses increases.

NOTE – The `minmisses` metric cannot be used for firewall load balancing, since the real server IP addresses used in calculating the score for this metric are different on each side of the firewall.

Hash

The hash metric uses IP address information in the client request to select a server. When selecting a server, a mathematical *hash* of the relevant IP address information is used as an index into the list of currently available servers. Any given IP address information will always have the same hash result, providing natural persistence, as long as the server list is stable.

The specific IP address information used depends on the application:

- For application redirection, the client destination IP address is used. All requests for a specific IP destination address will be sent to the same server. This is particularly useful for maximizing successful cache hits.
- For SLB, the client source IP address is used. All requests from a specific client will be sent to the same server. This option is useful for applications where client information must be retained between sessions.
- For FWLB, both the source and destination IP addresses are used to ensure that the two unidirectional flows of a given session are redirected to the same firewall. When selecting a server, a mathematical hash of the relevant IP address information is used as an index into the list of currently available servers. Any given IP address information will always have the same hash result, providing natural persistence, as long as the server list is stable. However, if a server is added to or removed from the group, then a different server might be assigned to a subsequent session with the same IP address information even though the original server is still available. Open connections are not cleared.

NOTE – The hash metric provides more distributed load balancing than *minmisses* at any given instant. It should be used if the statistical load balancing achieved using *minmisses* is not as optimal as desired. If the load balancing statistics with *minmisses* indicate that one server is processing significantly more requests over time than other servers, consider using the hash metric.

Least Connections

With the `leastconns` metric, the number of connections currently open on each real server is measured in real time. The server with the fewest current connections is considered to be the best choice for the next client connection request.

This option is the most self-regulating, with the fastest servers typically getting the most connections over time.

Round Robin

With the `roundrobin` metric, new connections are issued to each server in turn; that is, the first real server in the group gets the first connection, the second real server gets the next connection, followed by the third real server, and so on. When all the real servers in this group have received at least one connection, the issuing process starts over with the first real server.

Response Time

The response metric uses real server response time to assign sessions to servers. The response time between the servers and the switch is used as the weighting factor. The switch monitors and records the amount of time it takes for each real server to reply to a health check to adjust the real server weights. The weights are adjusted so they are inversely proportional to a moving average of response time. In such a scenario, a server with half the response time as another server will receive a weight twice as large.

NOTE – The effects of the response weighting apply directly to the real servers and are not necessarily confined to the real server group. When response time-metered real servers are also used in other real server groups that use the `leastconns` or `roundrobin` metrics, the response weights are applied on top of the `leastconns` or `roundrobin` calculations for the affected real servers. Since the response weight changes dynamically, this can produce fluctuations in traffic distribution for the real server groups that use the `leastconns` or `roundrobin` metrics.

Bandwidth

The bandwidth metric uses real server octet counts to assign sessions to a server. The switch monitors the number of octets sent between the server and the switch. Then, the real server weights are adjusted so they are inversely proportional to the number of octets that the real server processes during the last interval.

Servers that process more octets are considered to have less available bandwidth than servers that have processed fewer octets. For example, the server that processes half the amount of octets over the last interval receives twice the weight of the other servers. The higher the bandwidth used, the smaller the weight assigned to the server. Based on this weighting, the subsequent requests go to the server with the highest amount of free bandwidth. These weights are automatically assigned.

The bandwidth metric requires identical servers with identical connections.

NOTE – The effects of the `bandwidth` weighting apply directly to the real servers and are not necessarily confined to the real server group. When bandwidth-metered real servers are also used in other real server groups that use the `leastconns` or `roundrobin` metrics, the bandwidth weights are applied on top of the `leastconns` or `round-robin` calculations for the affected real servers. Since the bandwidth weight changes dynamically, this can produce fluctuations in traffic distribution for the real server groups that use the `leastconns` or `roundrobin` metrics.

Weights for Real Servers

Weights can be assigned to each real server. These weights can bias load balancing to give the fastest real servers a larger share of connections. Weight is specified as a number from 1 to 48. Each increment increases the number of connections the real server gets. By default, each real server is given a weight setting of 1. A setting of 10 would assign the server roughly 10 times the number of connections as a server with a weight of 1. To set weights, enter the following commands:

<code>>> # /cfg/slb/real <real server number></code>	<i>(Select the real server)</i>
<code>>> Real server# weight 10</code>	<i>(10 times the number of connections)</i>

NOTE – Weights are not applied when using the `hash` or `minmisses` metrics.

Connection Time-outs for Real Servers

In some cases, open TCP/IP sessions might not be closed properly (for example, the switch receives the `SYN` for the session, but no `FIN` is sent). If a session is inactive for 10 minutes (the default), it is removed from the session table in the switch. To change the time-out period, enter the following:

<code>>> # /cfg/slb/real <real server number></code>	<i>(Select the real server)</i>
<code>>> Real server# tmout 4</code>	<i>(Specify an even numbered interval)</i>

The example above would change the time-out period of all connections on the designated real server to four minutes.

Maximum Connections for Real Servers

You can set the number of open connections each real server is allowed to handle for SLB. To set the connection limit, enter the following:

```
>> # /cfg/slb/real <real server number>      (Select the real server)
>> Real server# maxcon 1600                  (Allow 1600 connections maximum)
```

Values average from approximately 500 HTTP connections for slower servers to 1500 for quicker, multiprocessor servers. The appropriate value also depends on the duration of each session and how much CPU capacity is occupied by processing each session. Connections that use a lot of Java or CGI scripts for forms or searches require more server resources and thus a lower maxcon limit. You may wish to use a performance benchmark tool to determine how many connections your real servers can handle.

When a server reaches its maxcon limit, the switch no longer sends new connections to the server. When the server drops back below the maxcon limit, new sessions are again allowed.

Backup/Overflow Servers

A real server can backup other real servers and can handle overflow traffic when the maximum connection limit is reached. Each backup real server must be assigned a real server number and real server IP address. It must then be enabled. Finally, the backup server must be assigned to each real server that it will back up. The following defines real server 4 as a backup for real servers 1 and 2:

```
>> # /cfg/slb/real 4                          (Select real server 4 as backup)
>> Real server 4# rip 200.200.200.5          (Assign backup IP address)
>> Real server 4# ena                         (Enable real server 4)
>> Real server 4# /cfg/slb/real 1             (Select real server 1)
>> Real server 1# backup 4                    (Real server 4 is backup for 1)
>> Real server 1# /cfg/slb/real 2             (Select real server 2)
>> Real server 2# backup 4                    (Real server 4 is backup for 2)
```

In a similar fashion, a backup/overflow server can be assigned to a real server group. If all real servers in a real server group fail or overflow, the backup comes online.

```
>> # /cfg/slb/group <real server group number> (Select real server group)
>> Real server group# backup r4               (Assign real server 4 as backup)
```

Real server groups can also use another real server group for backup/overflow:

```
>> # /cfg/slb/group <real server group number> (Select real server group)
>> Real server group# backup g2               (Assign real server group 2 as backup)
```

Extending SLB Topologies

For standard SLB, all client-to-server requests to a particular virtual server and all related server-to-client responses must pass through the same Alteon Application Switch. In complex network topologies, routers and other devices can create alternate paths around the Alteon Application Switch managing SLB functions (see [Figure 9-4 on page 142](#)). Under such conditions, the Alteon Application Switch with Alteon OS provides the following solutions:

- [Virtual Matrix Architecture](#)
- [Proxy IP Addresses](#)
- [Mapping Ports](#)
- [Direct Server Interaction](#)
- [Delayed Binding](#)

Virtual Matrix Architecture

Virtual Matrix Architecture (VMA) is a hybrid architecture that takes full advantage of the distributed processing capability in Alteon Application Switches. With VMA, the switch makes optimal use of system resources by distributing the workload to multiple processors, thereby improving switch performance and increasing session capacity. VMA also removes the topology constraints introduced by using Direct Access Mode (DAM). By default, VMA is enabled on the Alteon Application Switch (`/cfg/slb/adv/matrix`).

Proxy IP Addresses

In complex network topologies (see [Figure 9-4 on page 142](#)), SLB functions can be managed using a proxy IP address on the client switch ports.

When the client requests services from the switch's virtual server, the client sends its own IP address for use as a return address. If a proxy IP address is configured for the client port on the switch, the switch replaces the client's source IP address with the switch's own proxy IP address before sending the request to the real server. This creates the illusion that the switch originated the request.

The real server uses the switch's proxy IP address as the destination address for any response. SLB traffic is forced to return through the proper switch, regardless of alternate paths. Once the switch receives the proxied data, it puts the original client IP address into the destination address and sends the packet to the client. This process is transparent to the client.

NOTE – Because requests appear to come from the switch proxy IP address rather than the client source IP address, the network administrator should be aware of this behavior during debugging and statistics collection.

The proxy IP address can also be used for direct access to the real servers (see [“Direct Server Interaction” on page 163](#)).

Proxy IP Address and VMA

With VMA enabled, the concept of a global session table exists on the Alteon Application Switch. To identify which processor should process responses to proxied requests, a unique proxy IP address must be configured. If VMA is disabled, refer to [Table 9-4](#) to configure the corresponding proxy IP for the port and enable proxy.

[Table 9-4](#) shows how proxy IP addresses are shared among the FE and Gigabit ports on the Alteon 2424. For example, port numbers 1-12 and port 25 share proxy IP address 1. Alteon OS requires you to configure all four proxy IP addresses. The unused proxy IP addresses can be disabled using the command `/cfg/slb/port x/proxy dis`.

Table 9-4 Proxy IP addresses on Alteon Application Switch 2424

Proxy IP addresses	FE Port numbers	Gig port numbers
Proxy IP address 1	1 to 12	25
Proxy IP address 2	13 to 24	27
Proxy IP address 3	-	26
Proxy IP address 4	-	28

[Table 9-5](#) shows how proxy IP addresses are shared among the FE and Gigabit ports on the Alteon 2224.

Table 9-5 Proxy IP addresses on Alteon Application Switch 2224

Proxy IP addresses	FE Port numbers	Gig port numbers
Proxy IP address 1	1 to 12	25
Proxy IP address 2	13 to 24	-
Proxy IP address 3	-	26
Proxy IP address 4	-	-

Frames ingressing a port that has proxy option enabled (/cfg/slb/port x/proxy ena) are processed using one of the four configured proxy IP addresses. The source IP address determines which VMA switch processor to go to and the corresponding processor's proxy IP addresses replaces the packet's source IP address.

The source address is substituted with the proxy IP address of the port processing the request. If proxy is disabled, then the source IP address is not replaced with a proxy IP address.

```
>> # /cfg/slb/pip/pip1 10.10.10.10      (Configure a proxy IP address)
>> # /cfg/slb/port 1/proxy ena          (Enable proxy on port 1)
>> # /cfg/slb/port 2/proxy dis          (Disable proxy on port 2)
>> # /cfg/slb/port 3/proxy dis          (Disable proxy on port 3)
>> # /cfg/slb/port 4/proxy dis          (Disable proxy on port 4)
>> # /cfg/slb/port 5/proxy dis          (Disable proxy on port 5)
and so on....
```

The following procedure can be used for configuring proxy IP addresses:

1. Disable server processing on affected switch ports.

When implementing proxies, switch ports can be reconfigured to disable server processing. Referring to the [Table 9-2 on page 146](#), the following revised port conditions are used:

Table 9-6 Proxy Example: Port Usage

Port	Host	L4 Processing
1	Server A	None
2	Server B	None
3	Server C	None
4	Back-end NFS server provides centralized content for all three real servers. This port does not require switching features.	None
5	Client router A connects the switch to the Internet where all client requests originate.	Client
6	Client router B also connects the switch to the Internet where all client requests originate.	Client

The following commands are used to disable server processing on ports 1-3:

```
>> # /cfg/slb/port 1 (Select switch port 1)
>> SLB port 1# server dis (Disable server processing on port 1)
>> SLB port 1# ../port 2 (Select switch port 2)
>> SLB port 2# server dis (Disable server processing on port 2)
>> SLB port 2# ../port 3 (Select switch port 3)
>> SLB port 3# server dis (Disable server processing on port 3)
```

2. Configure a unique proxy address and enable proxy for the client ports.

Configure a unique proxy IP address and enable proxy on the *client* ports:

```
>> SLB port 6# ../pip (Select proxy IP menu)
>> Proxy IP Address# pip1 200.200.200.68 (Set proxy IP address)
>> # /cfg/slb/port 5/proxy ena (Select network port 5 and enable proxy)
>> SLB port 5# ../port 6 (Select network port 6)
>> SLB port 6# proxy ena (Enable proxy)
```

The proxies are transparent to the user.

3. (optional) If the Virtual Matrix Architecture (VMA) feature is enabled, configure all proxy IP addresses.

VMA is normally enabled on the switch. In addition to enhanced resource management, VMA eliminates many of the restrictions found in earlier versions of the Alteon OS. VMA does require, that you configure four unique proxy IP addresses.

The following commands can be used for configuring the additional proxy IP addresses:

```
>> SLB port 6# ../pip (Select the proxy IP address menu)
>> Proxy IP Address# pip2 200.200.200.69 (Set proxy IP address)
>> Proxy IP Address# pip3 200.200.200.70 (Set proxy IP address)
>> Proxy IP Address# pip4 200.200.200.71 (Set proxy IP address)
```

For information on using the VMA commands, see the *Alteon OS Command Reference* (/cfg/slb/adv/matrix) for more information.

4. Apply and save your changes.

NOTE – Remember that you must apply any changes in order for them to take effect, and you must save them if you wish them to remain in effect after switch reboot. Also, the /info/slb command is useful for checking the state of Server Load Balancing operations.

Mapping Ports

An Alteon Application Switch allows you to hide the identity of a port for security by mapping a virtual server port to a different real server port.

Mapping a Virtual Server Port to a Real Server Port

In addition to providing direct real server access in some situations (see [“Mapping Ports” on page 165](#)), mapping is required when administrators choose to execute their real server processes on different ports than the well-known TCP/UDP ports. Otherwise, virtual server ports are mapped directly to real server ports by default and require no mapping configuration.

Port mapping is configured from the Virtual Server Services menu. For example, to map the virtual server TCP/UDP port 80 to real server TCP/UDP port 8004, you could enter the following:

```
>> # /cfg/slb/virt 1/service 80 (Select virtual server port 80)
>> Virtual Server 1 http Service# rport 8004 (Map to real port 8004)
```

NOTE – If filtering is enabled, a proxy IP address is configured, or URL parsing is enabled on any switch port, then port mapping is supported with Direct Access Mode (DAM). For information about DAM, refer to [“Using Direct Access Mode” on page 164](#).

Mapping a Single Virtual Server Port to Multiple Real Server Ports

To take advantage of multi-CPU or multi-process servers, Alteon OS can be configured to map a single virtual port to multiple real ports. This capability allows the site managers, for example, to differentiate users of a service by using multiple service ports to process client requests.

An Alteon Application Switch supports up to 16 real ports per server when multiple `rports` are enabled. This feature allows the network administrator to configure up to 16 real ports for a single service port. This feature is supported in Layer 4 and Layer 7 and in cookie-based and SSL persistence switching environments.

When multiple real ports on each real server are mapped to a virtual port, the Alteon Application Switch treats the real server IP address/port mapping combination as a distinct real server.

NOTE – For each real server, you can only configure one service with multiple real ports.

Consider the following network:

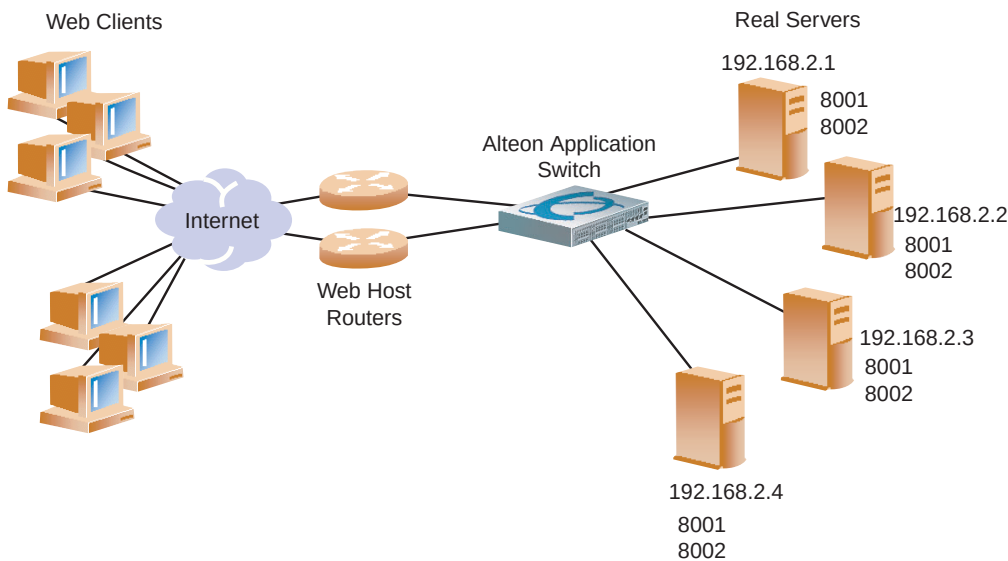


Figure 9-6 Basic Virtual Port to Real Port Mapping Configuration

Domain Name	Virtual Server IP Address	Ports Activated	Port Mapping	Real Server IP Address
www.right.com	192.168.2.100	80 (HTTP)	8001 (rport 1)	192.168.2.1 (RIP 1)
			8002 (rport 2)	192.168.2.2 (RIP 2)
				192.168.2.3 (RIP 3)
				192.168.2.4 (RIP 4)

In this example, four real servers are used to support a single service (HTTP). Clients access this service through a virtual server with IP address 192.168.2.100 on virtual port 80. Since each real server uses two ports (8001 and 8002) for HTTP services, the logical real servers are:

- 192.168.2.1/8001
- 192.168.2.1/8002
- 192.168.2.2/8001
- 192.168.2.2/8002
- 192.168.2.3/8001
- 192.168.2.3/8002
- 192.168.2.4/8001
- 192.168.2.4/8002

Load Balancing Metric

For each service, a real server is selected using the configured load balancing metric (hash, leastconns, minmisses, or roundrobin). To ensure even distribution, once an available server is selected, the switch will use the roundrobin metric to choose a real port to receive the incoming connection.

If the algorithm is `leastconns`, the switch sends the incoming connections to the logical real server (real server IP address/port combination) with the least number of connections.

The `/cfg/slb/virt` command defines the real server TCP or UDP port assigned to a service. By default, this is the same as the virtual port (service virtual port). If `rport` is configured to be different from the virtual port defined in `/cfg/slb/virt <virtual server number>/service <virtual port>`, the switch maps the virtual port to the real port.

NOTE – To use the single virtual port to multiple `rport` feature, configure this real server port option to be a value of 0. However, note that you *cannot* configure multiple services with multiple `rports` in the same server if the multiple `rport` feature is enabled.

Configuring Multiple Service Ports

Two commands, `addport` and `remport`, under the real server menu allow users to add or remove multiple service ports associated with a particular server. (A service port is a TCP or UDP port number.) For example: **`addport 8001`** and **`remport 8001`**.

1. Configure the real servers.

```
>> # /cfg/slb/real 1/rip 192.168.2.1/ena
>> # ../real 2/rip 192.168.2.2/ena
>> # ../real 3/rip 192.168.2.3/ena
>> # ../real 4/rip 192.168.2.4/ena
```

2. Add all four servers to a group.

```
>> # /cfg/slb/group 1
>> Real server Group 1# add 1
>> Real server Group 1# add 2
>> Real server Group 1# add 3
>> Real server Group 1# add 4
```

3. Configure a virtual server IP address.

```
>> # /cfg/slb/virt 1/vip 192.168.2.100/ena
```

4. Turn on multiple rport for Port 80.

```
>> # /cfg/slb/virt 1/service 80/rport 0
```

5. Add the ports to which the Web server listens.

```
>> # /cfg/slb/real 1/addport 8001      (Add port 8001 to real server 1)
>> # addport 8002                     (Add port 8002 to real server 1)
>> # ../real 2/addport 8001           (Add port 8001 to real server 2)
>> # addport 8002                     (Add port 8002 to real server 2)
>> # ../real 3/addport 8001           (Add port 8001 to real server 3)
>> # addport 8002                     (Add port 8002 to real server 3)
>> # ../real 4/addport 8001           (Add port 8001 to real server 4)
>> # addport 8002                     (Add port 8002 to real server 4)
```

Direct Server Interaction

Direct access to real servers can be provided in the following ways:

- [Using Direct Server Return](#)
- [Using Direct Access Mode](#)
- [Assigning Multiple IP Addresses](#)
- [Using Proxy IP Addresses](#)
- [Mapping Ports](#)
- [Monitoring Real Servers](#)

Using Direct Server Return

Some clients may need direct access to the real servers (for example, to monitor a real server from a management workstation). The Direct Server Return (DSR) feature allows the server to respond directly to the client. This capability is useful for sites where large amounts of data are flowing from servers to clients, such as with content providers or portal sites that typically have asymmetric traffic patterns.

DSR and content-intelligent Layer 7 switching cannot be performed at the same time because content intelligent switching requires that all frames go back to the switch for connection splicing.

NOTE – DSR requires that the server be set up to receive frames that have a destination IP address that is equal to the virtual server IP address.

The sequence of steps that are executed in this scenario are shown in [Figure 9-7](#):

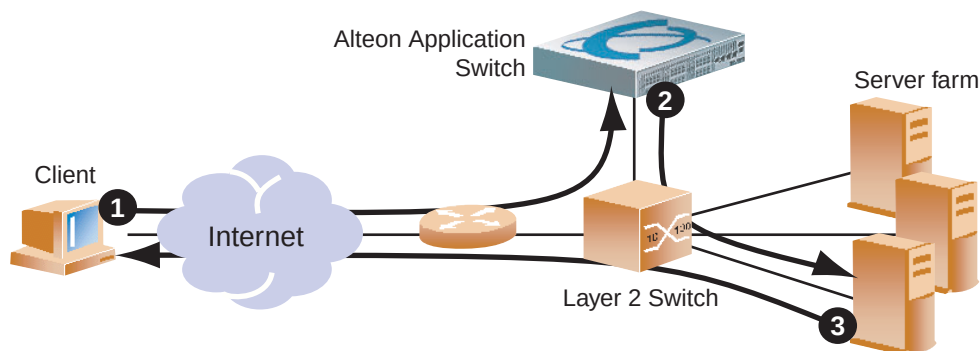


Figure 9-7 Direct Server Return

1. A client request is forwarded to the Alteon Application Switch.
2. Because only MAC addresses are substituted, the switch forwards the request to the best server, based on the configured load-balancing policy.
3. The server responds directly to the client, bypassing the switch, and using the virtual server IP address as the source IP address.

To set up DSR, use the following commands:

```
>> # /cfg/slb/real <real server number>/submac ena
>> # /cfg/slb/virt <virtual server number>/service <service number>/nonat ena
```

Using Direct Access Mode

When Direct Access Mode (DAM) (`/cfg/slb/direct`) is enabled on a switch, any client can communicate with any real server's load-balanced service. Also, with DAM enabled, any number of virtual services can be configured to load balance a real service.

Traffic sent directly to real server IP addresses is excluded from load-balancing decisions. The same clients may also communicate to the virtual server IP address for load-balanced requests.

NOTE – When DAM is enabled on a switch, port mapping and default gateway load balancing is supported only when filtering is enabled, a proxy IP address is configured, or URL parsing is enabled on any switch port.

Assigning Multiple IP Addresses

One way to provide both SLB access and direct access to a real server is to assign multiple IP addresses to the real server. For example, one IP address could be established exclusively for SLB and another could be used for direct access needs.

Using Proxy IP Addresses

Proxy IP addresses are used primarily to eliminate SLB topology restrictions in complex networks (see [“Proxy IP Addresses” on page 156](#)). Proxy IP addresses can also provide direct access to real servers.

If the switch is configured with proxy IP addresses and the client port is enabled for proxy, the client can access each real server directly using the real server’s IP address. The switch port must be connected to the real server and client processing must be disabled (see the `server` and `client` options under the `/cfg/slb/port` command in your *Alteon OS Command Reference*).

SLB is still accessed using the virtual server IP address.

Mapping Ports

When SLB is used without proxy IP addresses and without DAM, the Alteon Application Switch must process the server-to-client responses. If a client were to access the real server IP address and port directly, bypassing client processing, the server-to-client response could be mishandled by SLB processing as it returns through the Alteon Application Switch, with the real server IP address getting remapped back to the virtual server IP address on the Alteon Application Switch.

First, two port processes must be executed on the real server. One real server port will handle the direct traffic, and the other will handle SLB traffic. Then, the virtual server port on the Alteon Application Switch must be mapped to the proper real server port.

In [Figure 9-8](#), clients can access SLB services through well-known TCP port 80 at the virtual server’s IP address. The Alteon Application Switch behaving like a virtual server is mapped to TCP port 8000 on the real server. For direct access that bypasses the virtual server and SLB, clients can specify well-known TCP port 80 as the real server’s IP address.

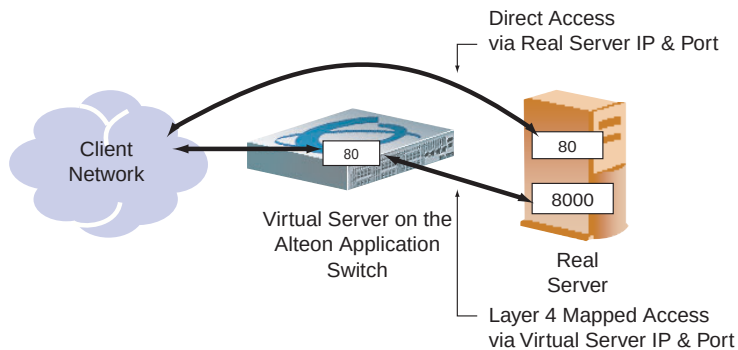


Figure 9-8 Mapped and Nonmapped Server Access

NOTE – Port mapping is supported with DAM when filtering is enabled, a proxy IP address is configured, or URL parsing is enabled on any switch port.

For more information on how to map a virtual server port to a real server port, see [“Mapping Ports” on page 160](#).

Monitoring Real Servers

Typically, the management network is used by network administrators to monitor real servers and services. By configuring the `mnet` and `mmask` options of the SLB Configuration Menu (`/cfg/slb/adv`), you can access the real services being load balanced.

NOTE – Clients on the management network do not have access to SLB services and cannot access the virtual services being load balanced.

The `mnet` and `mmask` options are described below:

- `mnet`: If defined, management traffic with this source IP address is allowed direct (non-SLB) access to the real servers. Only specify an IP address in dotted decimal notation. A range of IP addresses is produced when used with the `mmask` option.
- `mmask`: This IP address mask is used with `mnet` to select management traffic that is allowed direct real server access only.

Delayed Binding

The delayed binding feature on the switch prevents SYN Denial-of-Service (DoS) attacks on the server. DoS occurs when the server or switch is denied servicing the client because it is saturated with invalid traffic.

Typically, a three-way handshake occurs before a client connects to a server. The client sends out a synchronization (SYN) request to the server. The server allocates an area to process the client requests, and acknowledges the client by sending a SYN ACK. The client then acknowledges the SYN ACK by sending an acknowledgement (ACK) back to the server, thus completing the three-way handshake.

Figure 9-9 on page 167 illustrates a classic type of SYN DoS attack. If the client does not acknowledge the server's SYN ACK with a data request (REQ) and, instead, sends another SYN request, the server gets saturated with SYN requests. As a result, all of the servers resources are consumed and it can no longer service legitimate client requests.

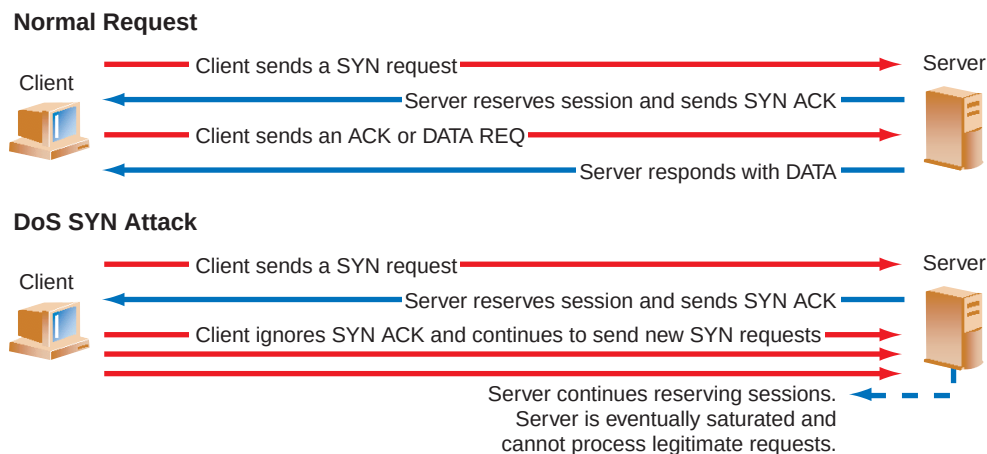
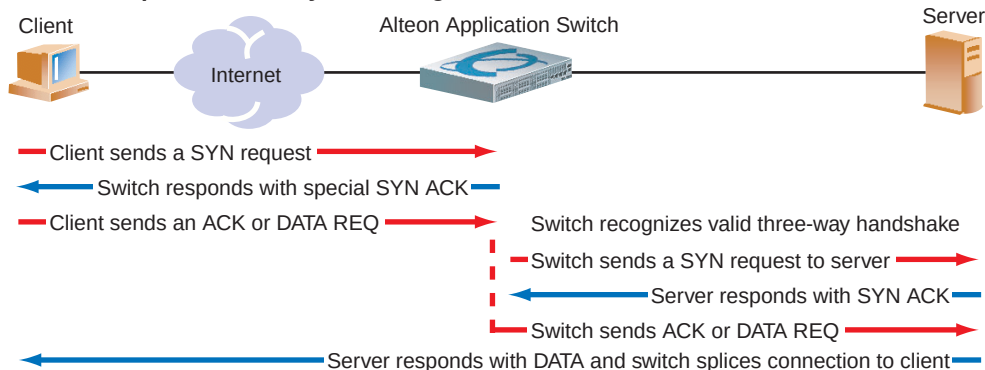
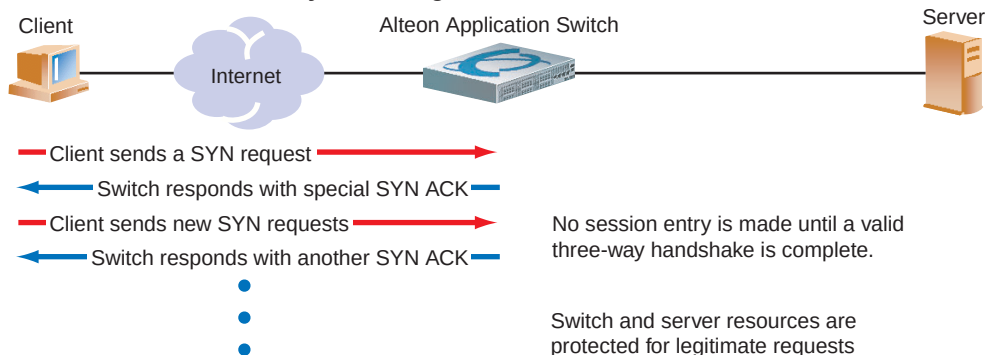


Figure 9-9 DoS SYN Attacks without Delayed Binding

Using an Alteon Application Switch with delayed binding, as illustrated in Figure 9-10 on page 168, the Alteon Application Switch intercepts the client SYN request before it reaches the server. The Alteon Application Switch responds to the client with a SYN ACK that contains embedded client information. The Alteon Application Switch does not allocate a session until a valid SYN ACK is received from the client or the three-way handshake is complete.

Normal Request with Delayed Binding**DoS SYN Attack with Delayed Binding****Figure 9-10** Repelling DoS SYN Attacks With Delayed Binding

Once the Alteon Application Switch receives a valid ACK or DATA REQ from the client, the Alteon Application Switch sends a SYN request to the server on behalf of the client, waits for the server to respond with a SYN ACK, and then forwards the clients DATA REQ to the server. Basically, the Alteon Application Switch delays binding the client session to the server until the proper handshakes are complete.

Thus, with delayed binding, two independent TCP connections span a session: one from the client to the Alteon Application Switch and the second from the switch to the selected server. The switch temporarily terminates each TCP connection until content has been received, thus preventing the server from being inundated with SYN requests.

NOTE – Delayed binding is automatically enabled when content intelligent switching features are used. However, if you are not parsing content, you must explicitly enable delayed binding if desired.

Configuring Delayed Binding

To configure your switch for delayed binding, use the following command:

```
>> # /cfg/slb/virt <virtual server number>/service <service type>/dbind
```

NOTE – Enable delayed binding without configuring any HTTP SLB processing or persistent binding types.

To configure delayed binding for cache redirection, see [“Delayed Binding for Cache Redirection” on page 248](#).

Detecting SYN Attacks

In Alteon OS, SYN attack detection is enabled by default, whenever delayed binding is enabled. SYN attack detection:

- Provides a way to track half open connections
- Activates a trap notifying that the configured threshold is exceeded
- Monitors DoS attacks and proactively signals alarm
- Provides enhanced security
- Improves visibility and protection for DoS attacks

The probability of a SYN attack is higher if excessive half-open sessions are being generated on the Alteon Application Switch. Half-open sessions show an incomplete three-way handshake between the server and the client. You can view the total number of half-open sessions from the `/stat/slb/layer7/maint` menu.

To detect SYN attacks, the Alteon Application Switch keeps track of the number of new half-open sessions for a set period of time. If the value exceeds the threshold, then a syslog message and an SNMP trap are generated.

You can change the default parameters for detecting SYN attacks in the `/cfg/slb/adv/synatk` menu. You can specify how frequently you want to check for SYN attacks, from 2 seconds to a minute and modify the default threshold representing the number of new half-open sessions per second.

Load Balancing Special Services

This section discusses load balancing based on special services, such as

- [IP Server Load Balancing](#)
- [FTP Server Load Balancing](#)
- [Domain Name Server \(DNS\) Load Balancing](#)
- [Real Time Streaming Protocol SLB](#)
- [Wireless Application Protocol SLB](#)
- [Intrusion Detection System Server Load Balancing](#)
- [WAN Link Load Balancing](#)

IP Server Load Balancing

IP server load balancing allows you to configure your Alteon Application Switch for server load balancing based on client's IP address only. Typically, the client IP address is used with the client port number to produce a session identifier. When the Layer 3 option is enabled, the switch uses only the client IP address as the session identifier.

To configure the switch for IP load balancing:

```
>> # /cfg/slb/virt <virtual server number>
>> Virtual Server 1# layr3 ena
>> Virtual Server 1# service ip
>> Virtual Server 1 IP Service# group <group number >
```

IP server load balancing must be used if IP traffic is totally encrypted and you do not have access to the content.

FTP Server Load Balancing

As defined in RFC 959, FTP uses two connections—one for control information and another for data. Each connection is unique. Unless the client requests a change, the server always uses TCP port 21 (a well-known port) for control information, and TCP port 20 as the default data port.

FTP uses TCP for transport. After the initial three-way handshake, a connection is established. When the client requests any data information from the server, it will issue a `PORT` command (such as `ls`, `dir`, `get`, `put`, `mget` and `mput`) via the control port.

There are two modes of FTP operation, active and passive:

- In *Active FTP*, the FTP server initiates the data connection.
- In *Passive FTP*, the FTP client initiates the data connection. Because the client also initiates the connection to the control channel, the passive FTP mode does not pose a problem with firewalls and is the most common mode of operation.

Alteon OS supports both active and passive modes of FTP operation. You can switch from active to passive or vice versa in the same FTP session.

FTP Network Topology Restrictions

FTP network topology restrictions are listed below:

- FTP control and data channels must be bound to the same real server.
- FTPP with port mapping is not supported.

Configuring FTP Server Load Balancing

1. **Make sure that a proxy IP address is enabled on the client port(s) or DAM is enabled.**
2. **Make sure the virtual port for FTP is set up for the virtual server.**

```
>> # /cfg/slb/virt 1/service ftp
```

3. **Enable FTP parsing on both services FTP and FTP-Data.**

```
>> Virtual Server 1 ftp Service# ftp ena
>> # /cfg/slb/virt 1/service 20/ftpp ena
```

4. **To make your configuration changes active, enter `apply` at any prompt in the CLI.**

```
>> Virtual Server 1 ftp Service# apply
```

You must `apply` and save any changes in order for them to take effect, and you must save them if you wish them to remain in effect after switch reboot.

Domain Name Server (DNS) Load Balancing

In Alteon OS, DNS load balancing allows you to choose the service based on the two forms of DNS queries: UDP and TCP. This enables the switch to send TCP DNS queries to one group of real servers and UDP DNS queries to another group of real servers. The requests are then load balanced among the real servers in that group.

Figure 9-11 shows four real servers load balancing UDP and TCP queries between two groups.

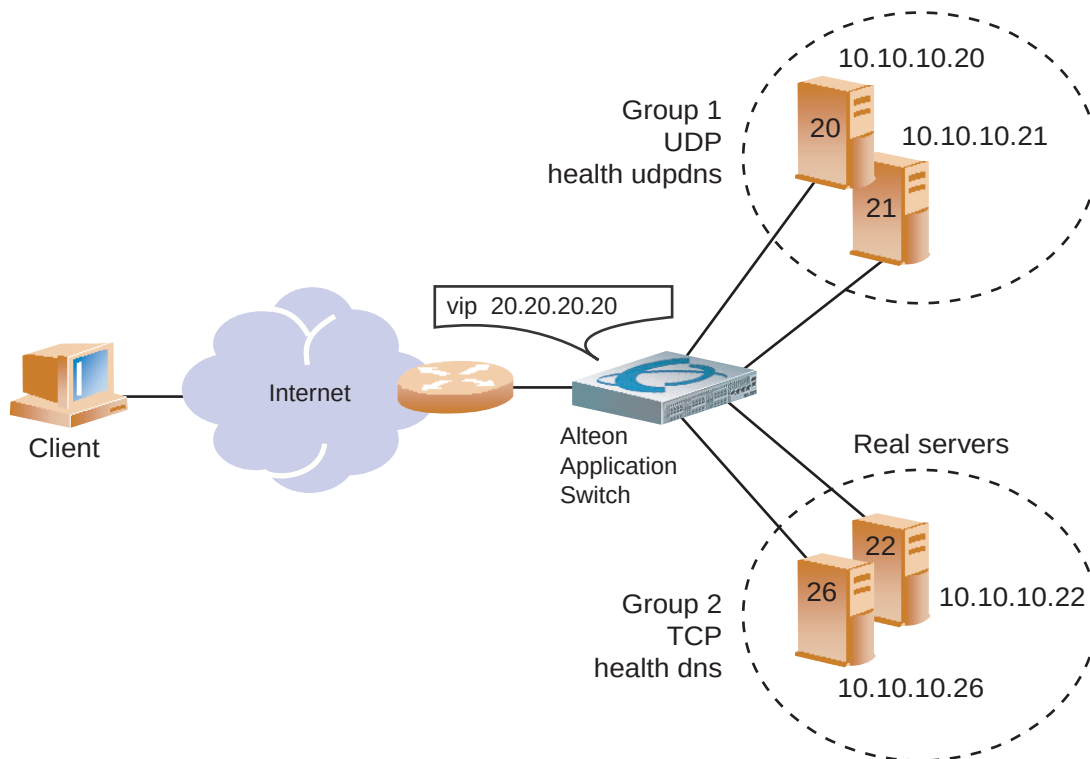


Figure 9-11 Layer 4 DNS Load Balancing

NOTE – You can configure both UDP and TCP DNS queries for the same virtual server IP address.

Preconfiguration Tasks

1. Enable server load balancing.

```
>> # /cfg/slb/on
```

2. Configure the four real servers and their real IP addresses.

```
>> # /cfg/slb/real 20
>> Real server 20 # ena                (Enable real server 20)
>> Real server 20 # rip 10.10.10.20    (Specify the IP address)
>> Real server 20 # ../real 21
>> Real server 21 # ena                (Enable real server 21)
>> Real server 21 # rip 10.10.10.21    (Specify the IP address)
>> Real server 20 # ../real 22
>> Real server 22 # ena                (Enable real server 22)
>> Real server 22 # rip 10.10.10.22    (Specify the IP address)
>> Real server 20 # ../real 26
>> Real server 26 # ena                (Enable real server 26)
>> Real server 26 # rip 10.10.10.26    (Specify the IP address)
```

3. Configure group 1 for UDP and group 2 for TCP.

```
>> # /cfg/slb/group 1                (Select real server group 1)
>> Real server group 1 # metric roundrobin (Specify the load balancing metric for group 1)

>> Real server group 1 # health udpdns    (Set the health check to UDP)
>> Real server group 1 # add 20           (Add real server 20)
>> Real server group 1 # add 21           (Add real server 21)
>> Real server group 1 # ../group 2
>> Real server group 2 # metric roundrobin (Specify the load balancing metric for group 2)

>> Real server group 2 # health dns        (Set the health check to TCP)
>> Real server group 2 # add 22           (Add real server 22)
>> Real server group 2 # add 26           (Add real server 26)
```

For more information on configuring health check, see [“UDP-Based DNS Health Checks” on page 269](#).

4. Define and enable the server ports and the client ports.

For more information, see [Step 6 “Define the port settings.” on page 146](#). Some DNS servers initiate upstream requests and must be configured both as server and client.

Configuring UDP-based DNS Load Balancing

1. **Configure and enable a virtual server IP address 1 on the switch.**

```
>> # /cfg/slb/virt 1/vip 20.20.20.20 (Specify the virt server IP address)
>> Virtual Server 1# ena (Enable the virtual server)
```

2. **Set up the DNS service for the virtual server, and add real server group 1.**

```
>> Virtual Server 1# service dns (Specify the DNS service)
>> Virtual Server 1 DNS Service# group 1 (Select the real server group)
```

3. **Disable delayed binding.**

Delayed binding is not required because UDP does not process session requests with a TCP three-way handshake.

```
>> Virtual Server 1 DNS Service# dbind dis (Disable delayed binding)
```

4. **Enable UDP DNS queries.**

```
>> Virtual Server 1 DNS Service# udp ena (Enable UDP balancing)
```

5. **Apply and save your configuration.**

```
>> Virtual Server 1 DNS Service# apply
>> Virtual Server 1 DNS Service# save
```

Configuring TCP-based DNS Load Balancing

1. **Configure and enable the virtual server IP address 2 on the switch.**

```
>> # /cfg/slb/virt 2/vip 20.20.20.20 (Specify the virt server IP address)
>> Virtual Server 2# ena (Enable the virtual server)
```

2. **Set up the DNS service for virtual server, and select real server group 2.**

```
>> Virtual Server 2# service dns (Specify the DNS service)
>> Virtual Server 2 DNS Service# group 2 (Select the real server group)
```

3. Enable delayed binding.

```
>> Virtual Server 2 DNS Service# dbind ena(Enable delayed binding)
```

4. As this is TCP-based load balancing, make sure to disable UDP DNS queries.

```
>> Virtual Server 2 DNS Service# udp dis (Disable UDP balancing)
```

5. Apply and save your configuration.

```
>> Virtual Server 2 DNS Service# apply  
>> Virtual Server 2 DNS Service# save
```

Real Time Streaming Protocol SLB

Real Time Streaming Protocol (RTSP) is an application-level protocol for control over the delivery of data with real-time properties as documented in RFC 2326. RTSP is the proposed standard for controlling streaming data over the Internet. RTSP uses RTP (Real-Time Transport Protocol) to format packets of multimedia content. RTSP is designed to efficiently broadcast audio-visual data to large groups.

Typically, a multimedia presentation consists of several streams of data (for example, video stream, audio stream, and text) that must be presented in a synchronized fashion. A multimedia client like Real Player or Quick Time Player downloads these multiple streams of data from the multimedia servers and presents them on the player screen.

RTSP is used to control the flow of these multimedia streams. Each presentation uses one RTSP control connection and several other connections to carry the audio/video/text multimedia streams. In this document, the term RTSP server refers to any multimedia server that implements the RTSP protocol for multimedia presentation.

How RTSP Server Load Balancing Works

The objective of RTSP server load balancing is to intelligently switch an RTSP request, and the other media streams associated with a presentation, to a suitable RTSP server based on the configured load-balancing metric. Typically, an RTSP client establishes a control connection to an RTSP server over TCP port 554 and the data flows over UDP or TCP.

Alteon OS supports two layer 7 metrics (URL hashing and URL pattern matching) and all layer 4 load-balancing metrics. This section discusses load balancing RTSP servers for layer 4; for information on load balancing RTSP servers for layer 7, see [“RTSP Load Balancing” on page 437](#).

For information on using RTSP with cache redirection, see [“RTSP Cache Redirection” on page 249](#).

NOTE – This feature is not applicable if the streaming media (multimedia) servers use HTTP protocol to tunnel RTSP traffic. To ensure that RTSP server load balancing works, make sure the streaming media server is configured for RTSP protocol.

Supported RTSP Servers

In a typical scenario, the RTSP client issues several sequences of commands to establish connections for each component stream of a presentation. There are several variations to this procedure, depending upon the RTSP client and the server involved. For example, there are two prominent RTSP server and client implementations.

The RTSP stream setup sequence is different for these two servers, and the switch handles each differently as shown below:

■ Real Server

Real Server from RealNetworks Corporation supports both UDP and TCP transport protocols for the RTSP streams. The actual transport is negotiated during the initialization of the connection. If TCP transport is selected, then all streams of data will flow in the TCP control connection itself. If UDP transport is chosen, the client and server negotiate a client UDP port, which is manually configurable.

The real media files that the Real Server plays have the extension “.rm”, “.ram” or “.smil”.

■ QuickTime Streaming Server

QuickTime Streaming Server from Apple Incorporated supports a QuickTime presentation that typically has two streams and therefore uses four UDP connections exclusively for transport and one TCP control connection. QuickTime clients use a UDP port, which is manually configurable. The QuickTime files have the extension “.mov”.

Alteon OS can also support other RTSP-compliant applications (excludes Windows Media Player because it is not RTSP compliant).

Configuring RTSP Load Balancing

In this example, the Alteon Application Switch is load balancing RTSP traffic between two media server farms as shown in [Figure 9-12](#). One group of media servers consist of QuickTime servers and the other group of servers consist of RealNetworks servers. Each group has its own virtual server IP address. For example, three Real Networks servers host media files for NortelNews; similarly, another three QuickTime servers host media files for AlteonNews. The content is duplicated among the servers in each group. Depending on the client request type, the Alteon Application Switch is configured to load balance in the following way:

- Retrieving files from the Real Networks server group

RTSP://www.NortelNews.com/*.ram, RTSP://www.NortelNews.com/*.rm, and RTSP://www.NortelNews.com/*.smil are load balanced among the Real Networks media servers using virtual IP address 30.30.30.100.

- Retrieving files from the QuickTime server group

RTSP://www.AlteonNews.com/*.mov is load balanced among the Quick Time media servers using virtual IP address 40.40.40.100.

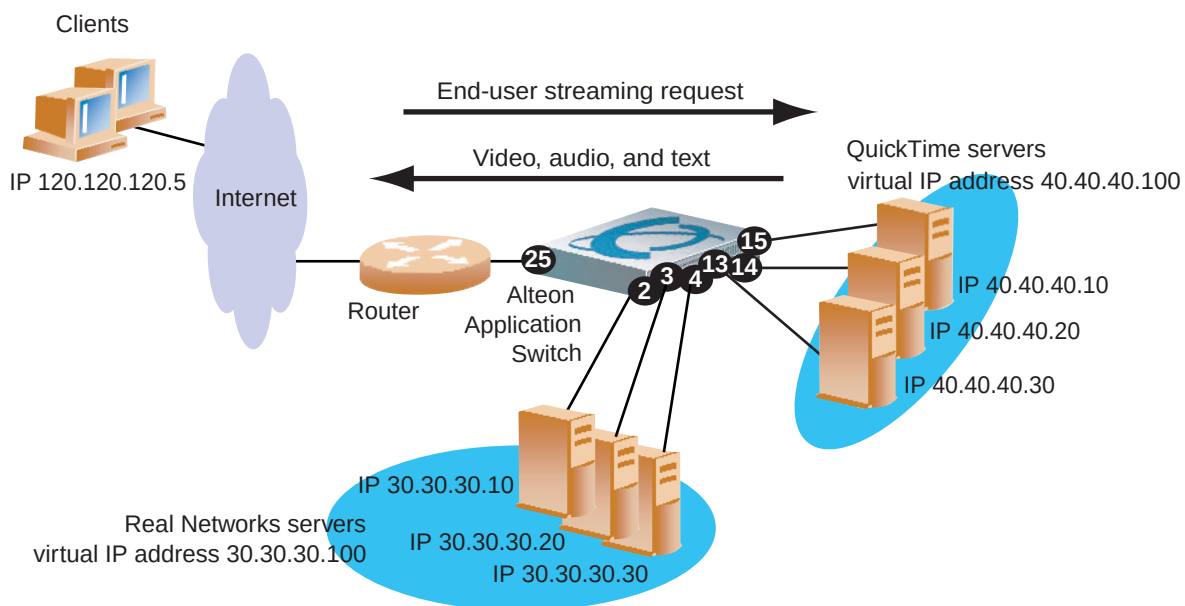


Figure 9-12 Load Balancing RTSP Servers

Follow this procedure to configure the topology illustrated in [Figure 9-12](#):

1. At the Alteon Application Switch, before you start configuring for RTSP load balancing:

- Connect each QuickTime server to the layer 2 switch
- Connect each RealNetworks server to the layer 2 switch
- Configure the IP addresses on all devices connected to the Alteon Application Switch
- Configure the IP interfaces on the switch
- Enable Virtual Matrix Architecture (VMA)
- Enable Direct Access Mode (DAM)
- Disable Bandwidth Management
- Disable proxy IP addressing

2. Enable server load balancing.

```
>> # /cfg/slb/on
```

3. Configure IP addresses for the real servers.

```
>> # /cfg/slb/real 1/rip 30.30.30.10/ena (Define IP address for Real server 1)
>> # /cfg/slb/real 2/rip 30.30.30.20/ena (Define IP address for Real server 2)
>> # /cfg/slb/real 3/rip 30.30.30.30/ena (Define IP address for Real server 3)
>> # /cfg/slb/real 4/rip 40.40.40.10/ena (Define IP address for Real server 4)
>> # /cfg/slb/real 5/rip 40.40.40.20/ena (Define IP address for Real server 5)
>> # /cfg/slb/real 6/rip 40.40.40.30/ena (Define IP address for Real server 6)
```

4. Create a group to support RealNetworks servers.

```
>> # /cfg/slb/group 100 (Define a group)
>>Real Server Group 100# add 1 (Add real server 1)
>>Real Server Group 100# add 2 (Add real server 2)
>>Real Server Group 100# add 3 (Add real server 3)
```

5. Create a group to support QuickTime servers.

```
>> # /cfg/slb/group 200 (Define a group)
>>Real Server Group 200# add 4 (Add real server 4)
>>Real Server Group 200# add 5 (Add real server 5)
>>Real Server Group 200# add 6 (Add real server 6)
```

6. Create a virtual server for the RealNetworks servers.

To configure a virtual server for layer 4 load balancing of RTSP, select **rtsp** or port **554** as a service for the virtual server.

```
>> # /cfg/slb/virt 1 (Select the virtual server)
>>Virtual Server 1# vip 30.30.30.100 (Set IP address for the virtual server)
>>Virtual Server 1# service 554 (Add the RTSP service for the virtual server)
>>Virtual Server 1 rtsp Service# group 100 (Set the real server group)
>>Virtual Server 1 rtsp Service# ../ena (Enable virtual server)
```

7. Create a virtual server for the QuickTime servers.

To configure a virtual server for Layer 4 load balancing of RTSP, select **rtsp** or port **554** as a service for the virtual server.

```
>> # /cfg/slb/virt 2 (Select the virtual server)
>>Virtual Server 2# vip 40.40.40.100 (Set IP address for the virtual server)
>>Virtual Server 2# service 554 (Add the RTSP service for the virtual server)
>>Virtual Server 2 rtsp Service# group 200 (Set the Quicktime server group)
>>Virtual Server 2 rtsp Service# ../ena (Enable virtual server)
```

8. Enable server and client processing at the port level.

```
>> # /cfg/slb/port 25 (Select the client port)
>>SLB port 25# client ena (Enable client processing)
>>SLB port 1# ../port 2 (Select the server port)
>>SLB port 2# server ena (Enable server processing)
>>SLB port 2# ../port 3 (Select the server port)
>>SLB port 3# server ena (Enable server processing)
>>SLB port 3# ../port 4 (Select the server port)
>>SLB port 4# server ena (Enable server processing)
>>SLB port 4# ../port 13 (Select the server port)
>>SLB port 13# server ena (Enable server processing)
>>SLB port 13# ../port 14 (Select the server port)
>>SLB port 14# server ena (Enable server processing)
>>SLB port 14# ../port 15 (Select the server port)
>>SLB port 15# server ena (Enable server processing)
```

9. Apply and save your configuration.

```
>> SLB port 15# apply
>> SLB port 15# save
```

Clients retrieving files of type `RTSP://nortelnews.com/headlines.ram` use virtual IP address 30.30.30.100 of the RealNetworks server group and clients retrieving files of type `RTSP://nortelnews.com/headlines.mov` use virtual IP address 40.40.40.100 of the QuickTime server group.

Wireless Application Protocol SLB

Wireless Application Protocol (WAP) is an open, global specification for a suite of protocols designed to allow wireless devices to communicate and interact with other devices. It empowers mobile users with wireless devices to easily access and interact with information and services instantly by allowing non-voice data, such as text and images, to pass between these devices and the Internet. Wireless devices include cellular phones, pagers, Personal Digital Assistants (PDAs), and other hand-held devices.

WAP supports most wireless networks and is supported by all operating systems—with the goal of inter-operability. A WAP Gateway translates Wireless Markup Language (WML)—which is a WAP version of HTML—into HTML/HTTP so that requests for information can be serviced by traditional Web servers.

To load balance WAP traffic among available parallel servers, the switch must provide *persistency* so that the clients can always go to the same WAP gateway to perform WAP operation.

Figure 9-13 shows the user is first authenticated by the remote access server. In this example the RADIUS servers are integrated with the WAP gateways.

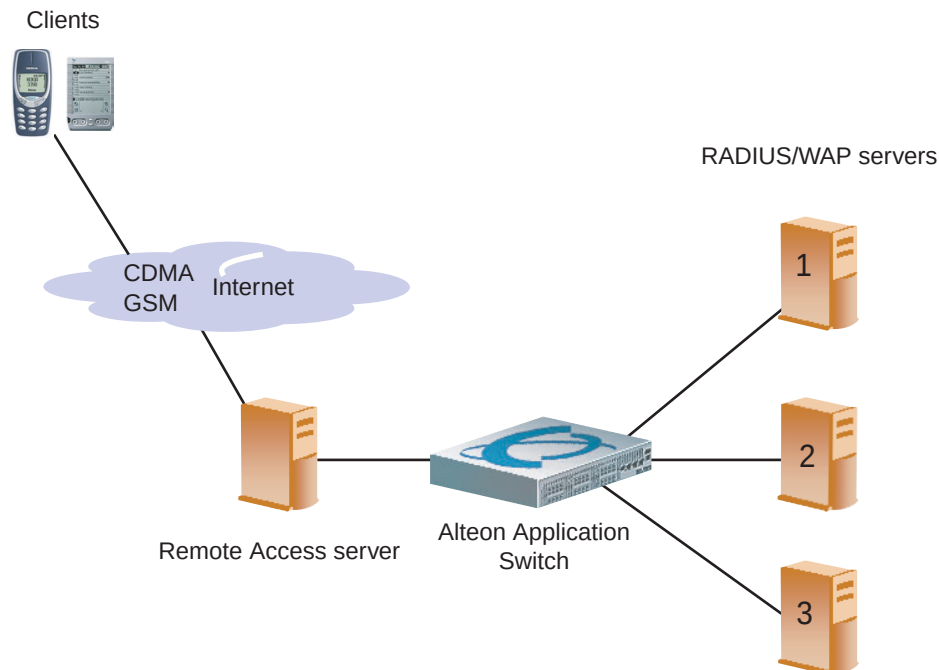


Figure 9-13 Load Balancing WAP Gateways

Alteon OS allows you to configure the Alteon Application Switch to select a WAP gateway for each client request based on one of the following two methods: static session entry via TPCP or RADIUS snooping.

The following topics are discussed in this section:

- [WAP SLB Using RADIUS Static Session Entries](#)
- [WAP SLB Using RADIUS Snooping](#)
- [Configuring for WAP Load Balancing](#)

WAP SLB Using RADIUS Static Session Entries

RADIUS, a proposed IETF standard is a client/server protocol that enables remote access servers to communicate with a central server to authenticate dial-in users and authorize their access to the requested network or service. RADIUS allows a company to maintain user profiles in a central database that all remote servers can share. It provides better security, allowing a company to set up a policy that can be applied at a single administered network point.

The RADIUS server uses a static session entry to determine which real WAP gateway should receive the client sessions. Typically, each WAP gateway is integrated with a RADIUS server on the same host, and a RADIUS request packet is allowed to go to any of the RADIUS servers. Upon receiving a request from a client, the RADIUS server instructs the switch to create a static session entry in the switch via Transparent Proxy Control Protocol (TPCP). TPCP is an Alteon proprietary protocol that is used to establish communication between the RADIUS servers and the Alteon Application Switch. It is UDP-based and uses ports 3121, 1812, and 1645.

The RADIUS servers use TPCP to add or delete static session entries on the switch. Typically, a regular session entry is added or removed by the switch itself. A static session entry, like a regular session entry, contains information such as the client IP address, the client port number, real port number, virtual (destination) IP address, virtual (destination) port number.

A static session entry added via TPCP to the switch does not age out. The entry will only be deleted by another TPCP `Delete Session` request. If the user adds session entries using the traditional server load balancing methods, the entries will continue to age out.

Because TPCP is UDP-based, the `Add/Delete Session` requests may get lost during transmission. The WAP gateway issues another `Add Session` request on detecting that it has lost a request. The WAP gateway detects this situation when it receives WAP traffic that does not belong to that WAP gateway. If a `Delete Session` request is lost, it will be overwritten by another `Add Session` request.

How WAP SLB Works Using Static Session Entries

1. **On dialing, the user is first authenticated by the Remote Access Server (RAS).**
2. **The RAS sends a RADIUS authentication request to one of the RADIUS servers, which can be integrated with a WAP gateway.**
3. **If the user is accepted, the RADIUS server determines which WAP gateway is right for this user and informs the switch of the decision via TPCP.**
4. **The switch receives a request from the RADIUS server and adds a session entry to its session table to bind a WAP gateway with that user.**
5. **A response packet is sent back to the RAS by the RADIUS server.**
6. **The RAS receives the packet and allows the WAP traffic for that user.**
7. **If the user disconnects, the RAS detects it and sends this information to the RADIUS server.**
8. **The RADIUS server removes the session entry for that user.**

WAP SLB Using RADIUS Snooping

RADIUS snooping is similar to the static session entry method in the way that a static session entry is added to (or removed from) the switch for the WAP traffic for a user; it is different from the static session entry method in the way that RADIUS accounting packets are snooped by the Alteon switch instead of by the RADIUS server using TPCP.

Radius snooping allows the Alteon Application Switch to examine RADIUS accounting packets for client information. This information is needed to add to or delete static session entries in the switch's session table so that it can perform the required persistency for load balancing. A static session entry does not age out. Such an entry, added using RADIUS snooping, will only be deleted using RADIUS snooping. The switch load balances both the RADIUS and WAP gateway traffic using the same virtual server IP address.

How WAP SLB Works Using RADIUS Snooping

Before the RAS allows the WAP traffic for a user to pass in and out of the gateway, it sends a `RADIUS Accounting Start` message to one of the RADIUS servers. The switch then *snoops* on the packet to extract the required information. It needs to know the type of the `RADIUS Accounting` message, the client IP address, the caller ID, and the user's name. If it finds this information, the switch adds a session entry to its session table. If any of this information is missing, the switch will not take any action to handle the session entry.

When the client ends the WAP connection, RAS sends `RADIUS Accounting Stop` packet. If the switch finds the needed information in a `RADIUS Accounting Stop` packet, it removes the corresponding session entry from its table. The following steps occur for RADIUS snooping:

1. **The user is authenticated on dialing.**
2. **The RAS establishes a session with the client and sends a `RADIUS Accounting Start` message with the client IP address to the RADIUS server.**
3. **The switch snoops on the RADIUS accounting packet and adds a session entry if it finds enough information in the packet.**
4. **The switch load balances the WAP traffic to a specific WAP gateway.**
5. **When the client terminates the session, the RAS sends an `Accounting Stop` message to the RADIUS server, and the session entry is deleted from the switch.**

Consider the following items before configuring RADIUS snooping:

- The same virtual server IP address must be used when load balancing both the RADIUS accounting traffic and WAP traffic.
- All the RADIUS servers must use the same UDP port for RADIUS accounting services.

- Before a session entry is recorded on the switch, WAP packets for a user can go to any of the real WAP gateways.
- If a session entry for a client cannot be added because of resource constraints, the subsequent WAP packets for that client will not be load balanced correctly. The client will need to drop the connection and then reconnect to the wireless service.
- The persistence of a session cannot be maintained if the number of healthy real WAP gateways changes during the session. For example, if a new WAP server comes into service or some of the existing WAP servers are down, the number of healthy WAP gateway changes and, in this case, the persistence for a user cannot be maintained.
- Persistence cannot be maintained if the user moves from one ISP to another, or if the base of the user's session changes (that is, from `CALLING_STATION_ID` to `USER_NAME`, or vice versa). For example, if a user moves out of a roaming area, it is possible that his/her `CALLING_STATION_ID` is not available in the RADIUS Accounting packets. In such a case, the switch uses `USER_NAME` to choose a WAP server instead of `CALLING_STATION_ID`. Thus, persistence cannot be maintained.

Configuring for WAP Load Balancing

Follow this procedure to configure the topology illustrated in [Figure 9-13](#):

1. Before you start configuring for WAP load balancing:

- Enable layer 3 server load balancing.

```
>> # /cfg/slb/virt <number>/layr3 ena
```

- Enable UDP under the WAP services (ports 9200 to 9203) menu.

```
>> # /cfg/slb/virt <number>/service <name|number>/udp ena
```

- Configure for RADIUS service 1812 and 1645.

NOTE – The radius service number specified on the switch must match with the service specified on the server.

2. At the Alteon Application Switch, configure the switch for basic SLB.

```
>> # /cfg/slb/on
```

3. Configure IP addresses for the RADIUS/WAP Gateways.

```
>> # /cfg/slb/real 1/rip 1.1.1.100      (Define address for WAP Gateway1)
>> Real server 1# ena                  (Enable real server 1)
>> # /cfg/slb/real 2/rip 2.2.2.100      (Define address for WAP Gateway 2)
>> Real server 2# ena                  (Enable real server 2)
>> # /cfg/slb/real 3/rip 3.3.3.100      (Define address for WAP Gateway 3)
>> Real server 3# ena                  (Enable real server 3)
```

4. Create a group to load balance the WAP Gateways.

```
>> # /cfg/slb/group 100                (Define a group)
>>Real Server Group 100# add 1          (Add real server 1)
>>Real Server Group 100# add 2          (Add real server 2)
>>Real Server Group 100# add 3          (Add real server 3)
```

5. Enable the external notification from WAP gateway to add and delete session request if you are using static session via TPCP.

```
>> # cfg/slb/adv/tpcp ena
```

6. Enable TPCP for adding and deleting WAP sessions.

```
>> # cfg/slb/wap/tpcp ena
```

7. Apply and save your configuration.

```
>> Filter 1 Advanced# apply
>> Filter 1 Advanced# save
```

Continue with the procedure (Steps 8–10) if you are configuring for RADIUS snooping.

8. (optional) Configure the following filter on the switch to examine a RADIUS accounting packet. Set the basic filter parameters.

>> # /cfg/slb/filt 1	(Select the filter)
>> Filter 1 # dip 10.10.10.100	(Set the destination IP address)
>> Filter 1 # dmask 255.255.255.255	(Set the destination IP mask)
>> Filter 1 # proto udp	(Set the protocol to UDP)
>> Filter 1 # dport 1813	(Set the destination port)
>> Filter 1 # action redir	(Set the action to redirect)
>> Filter 1 # group 1	(Set the group for redirection)
>> Filter 1 # rport 1813	(Set server port for redirection)

9. Turn on proxy and RADIUS snooping.

>> Filter 1 # adv	(Select the advance filter menu)
>> Filter 1 Advanced# proxy ena	(Enable proxy)
>> Filter 1 Advanced# rdsnp ena	(Enable RADIUS snooping)

10. Apply and save your configuration.

>> Filter 1 Advanced# apply
>> Filter 1 Advanced# save

NOTE – Alteon OS supports Virtual Router Redundancy Protocol (VRRP) and stateful failover, using both static session entries and RADIUS snooping. However, active-active configuration with Stateful Failover is not supported.

Intrusion Detection System Server Load Balancing

Intrusion Detection System (IDS) is a type of security management system for computers and networks. An Intrusion Detection System gathers and analyzes information from various areas within a computer or a network to identify possible security breaches, which include both intrusions (attacks from outside the organization) and misuse (attacks from within the organization).

Intrusion detection functions include:

- Monitoring and analyzing both user and system activities
- Analyzing system configurations and vulnerabilities
- Assessing system and file integrity
- Recognizing patterns typical of attacks
- Analyzing abnormal activity patterns
- Tracking user policy violations

Intrusion detection devices inspect every packet before it enters a network, looking for any signs of an attack. The attacks are recorded and logged in an attempt to guard against future attacks and to record the information about the intruders.

IDS server load balancing helps scale intrusion detection systems since it is not possible for an individual server to scale information being processed at gigabit speeds.

How Intrusion Detection Server Load Balancing Works

Alteon OS allows the switch to forward a copy of the IP packets to an Intrusion Detection server. IDS SLB must be enabled on the incoming ports and enabled for the groups containing the IDS real servers. The IDS SLB-enabled switch copies packets entering IDS-enabled ports that create an SLB session on the switch to a group of intrusion detection servers. The IDS server is selected based on the IDS group metric.

Configuring IDS Server Load Balancing

The following list summarizes the primary steps involved in configuring IDS load balancing:

- Set up the IDS servers

Determine if you want to setup the IDS servers in stealth mode (without IP addresses) or with non-routable IP addresses.

- Create the IDS groups

Create real server groups for the IDS servers. You may create multiple IDS groups to segregate incoming traffic based on protocols.

- ☐ Choose the metric for the group: `hash` or `minmisses`
- ☐ Choose the health check for the group: `link`, `icmp`, or `arp`
- ☐ Enable IDS on the group
- ☐ Select the type of traffic that is captured by the group by defining the IDS `rport` value. (The default for IDS `rport` is `any`).

If multiple groups are configured for the same `rport` then only ONE of the groups will be used for server load balancing.

- Enable IDS on the incoming ports (both client and server ports)

Enabling IDS at the port level enables the Alteon Application Switch to make a copy of the frames ingressing the port and forward the copy to the IDS server group.

- Configure filter processing on the incoming ports with the IDS hash metric.

This allows a session entry to be created for frames ingressing the port. IDS load balancing requires a session entry to be created in order to store the information regarding which IDS server to send the request.

If the allow filter is configured to hash on both the client and server IP address, then this will ensure that both client and server traffic belonging to the same session is sent to the same IDS server. For more information, see [“Example 2: Load Balancing to Multiple IDS Groups” on page 194](#). If the port is configured for client processing only, then the switch hashes on the source IP address only.

The following two examples illustrate IDS load balancing in two different network environments. In [Example 1: Load Balancing to a Single IDS Group](#), one switch is dedicated to load balancing two IDS servers in a single group and a second switch is performing standard server load balancing. In [Example 2: Load Balancing to Multiple IDS Groups](#), a single switch is performing both IDS load balancing and standard server load balancing. In example 2, two IDS groups are configured—IDS group 51 is for HTTP traffic only and IDS group 52 is for all other traffic.

Example 1: Load Balancing to a Single IDS Group

Figure 9-14 illustrates a basic configuration for load balancing client and server traffic to the IDS servers. Alteon Application Switch 1 is performing IDS load balancing and Alteon Application Switch 2 is performing standard server load balancing. IDS is enabled on the client port (port 25) and both the firewall ports (ports 26 and 27).

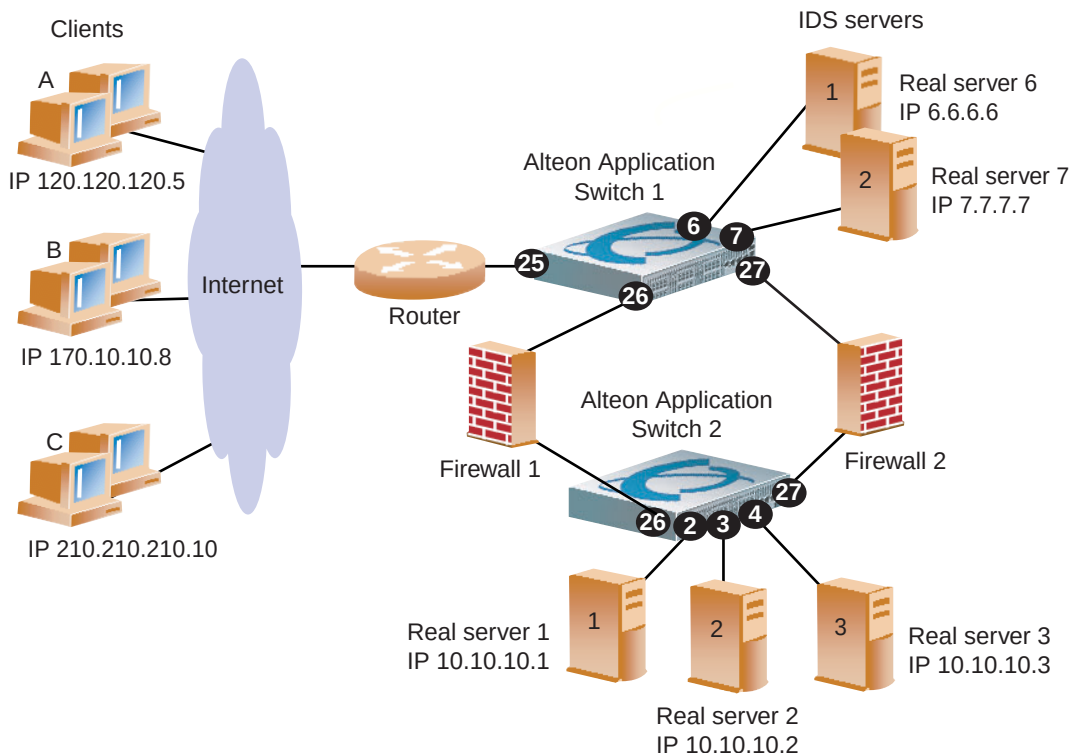


Figure 9-14 Server load balancing and IDS Load balancing

When the client request enters port 25 on Alteon Application Switch 1, the switch makes a copy of the packet. The switch load balances the copied packet between the two IDS servers based on the configured load balancing metric (hash or minmisses). The original data packet however, enters Alteon Application Switch 2 through the firewall and Alteon Application Switch 2 performs standard server load balancing on the client data between the three real servers. The client request is processed and returned to Alteon Application Switch 1 via the firewall. An allow filter at port 26 and port 27 causes the Alteon Application Switch to make a copy of the request and directs the copy to the IDS server group.

Follow this procedure to configure the topology illustrated in [Figure 9-14](#):

1. **Set up the IDS servers.**

To configure the IDS servers as real servers you must consider the setup of the IDS servers and the selection of the health check. Typically, most IDS servers are setup in stealth mode (without IP addresses); but, they can also be set up with non-routable IP addresses.

Table 9-7 Setting Up IDS Servers

IP Addresses for IDS Servers	Health Check Type	Port Configuration	Explanation
Stealth mode (without IP addresses or dummy IP addresses)	Link	–IDS servers must directly connect to separate physical ports on the switch. –Real server # of IDS server must match the physical port # (1 to 26) on the switch	To send packets to different IDS servers you must connect IDS servers to separate switch ports or associate them with different VLANs and tag the packets accordingly. Because unmodified frames are sent to the IDS servers, the switch does not use the L2 destination field of the packet to direct it to the correct IDS server. Basically, the switch port or the VLAN tag is used to identify the destination IDS server. However, if the ingress packet is already tagged, then you must use different switch ports for different IDS servers.
With IP addresses	ICMP or ARP	IDS servers need not be directly connected to the application switch. The IDS servers may be connected via a layer 2 switch.	The data packet is modified, so that it is addressed to the IDS servers. Destination MAC address is changed to Real server MAC address.

2. **At the Alteon Application Switch, configure the IDS servers as real servers.**

In [Figure 9-14](#) the IDS servers are configured in stealth mode. Match the real server number with the physical port number to which the IDS servers are connected, and configure dummy IP addresses. The real servers must be numbered between 1-63.

```
>> # /cfg/slb/real 6/rip 6.6.6.6/ena
>> # /cfg/slb/real 7/rip 7.7.7.7/ena
```

(Define a dummy IP address for IDS server 6)

(Define a dummy IP address for IDS server 7)

3. Create a group and add the IDS servers.

The group must be numbered between 1–63.

```
>> # /cfg/slb/group 50                (Define a group)
>>Real Server Group 50# add 6          (Add IDS server 6)
>>Real Server Group 50# add 7          (Add IDS server 7)
```

4. Define the group metric for the IDS server group.

IDS server load balancing supports hash and minmisses metrics only.

```
>>Real Server Group 50# metric hash    (Set the metric to hash)
```

5. Define the health check for the group.

Configure link health check which is specifically developed for IDS servers set up in stealth mode (without IP addresses).

```
>>Real Server Group 50# health link    (Set the health check to link)
```

6. Define the group for IDS server load balancing.

```
>>Real Server Group 50# idslb ena      (Enable IDS for the server group)
```

7. Select the rport for the IDS group.

```
>>Real Server Group 50# idsrprt any
```

8. Enable IDS on the client and server ports.

This enables frames ingressing the port to be copied to the IDS servers.

```
>># /cfg/slb/port 25/ids ena           (Enable IDS processing for port 25)
>>SLB port 25# ../port 26/ids ena      (Enable IDS processing for port 26)
>>SLB port 26# ../port 27/ids ena      (Enable IDS processing for port 27)
```

In addition to enabling IDS at the port level, a filter must be configured to create a session entry for non-SLB frames ingressing the port. IDS load balancing requires a session entry to be created to store the information regarding which IDS server to send to.

9. Create an allow filter and configure the filter with the `idshash` metric.

```
>> # /cfg/slb/filt 2048                (Select the menu for Filter 2048)
>> Filter 2048# sip any                (From any source IP address)
>> Filter 2048# dip any                (To any destination IP address)
>> Filter 2048# action allow           (Allow matching traffic to pass)
>> Filter 2048# ena                   (Enable the filter)
>> Filter 2048# adv/idshash both       (Set the hash metric parameter)
```

The IDS hash metric is set to hash on both the source and destination IP addresses. Hashing on both source and destination IP address ensures that the returning traffic goes to the same IDS server. If the port is configured for client processing only, then the switch hashes on the source IP address. By default, the IDS hash metric hashes on the source IP address only.

10. Apply the allow filter to ports 25, 26, and 27.

The allow filter must be applied on all ports that require layer 4 traffic to be routed to the IDS servers.

```
>> Filter 2048# ../port 25             (Select the client port)
>> SLB Port 25# add 2048               (Apply the filter to the client port)
>> SLB Port 25# filter ena             (Enable the filter)
>> SLB Port 25# ../port 26             (Select port 26)
>> SLB Port 26# add 2048               (Apply the filter to port 26)
>> SLB Port 26# filter ena             (Enable the filter)
>> SLB Port 26# ../port 27             (Select port 27)
>> SLB Port 27# add 2048               (Apply the filter to port 27)
>> SLB Port 27# filter ena             (Enable the filter)
```

All ingressing traffic at these ports that match any of the filters configured for that port will be load balanced to the IDS groups. The allow filter is used at the end of the filter list to make sure that all traffic matches a filter. A *deny all* filter could also be used as the final filter instead of an *allow all* filter.

11. Apply and save your changes.

```
>> SLB Port 25# apply
>> SLB Port 25# save
```

12. Configure Alteon Application Switch 2 to load balance the real servers as described in “Configuring Server Load Balancing” on page 144.

- Configure the IP interfaces on the switch
- Configure the SLB real servers and add the real servers to the group
- Configure the virtual IP address
- Configure the SLB metric
- Enable SLB

A copy of layer 4 traffic from clients A, B, and C and from the real servers are directed to the IDS servers and load balanced between IDS servers 6 and 7.

Example 2: Load Balancing to Multiple IDS Groups

Figure 9-15 illustrates a single Alteon Application Switch performing both standard server load balancing and IDS server load balancing. In this example, two IDS groups are configured—IDS group 51 is for HTTP traffic only and IDS group 52 is for all other traffic.

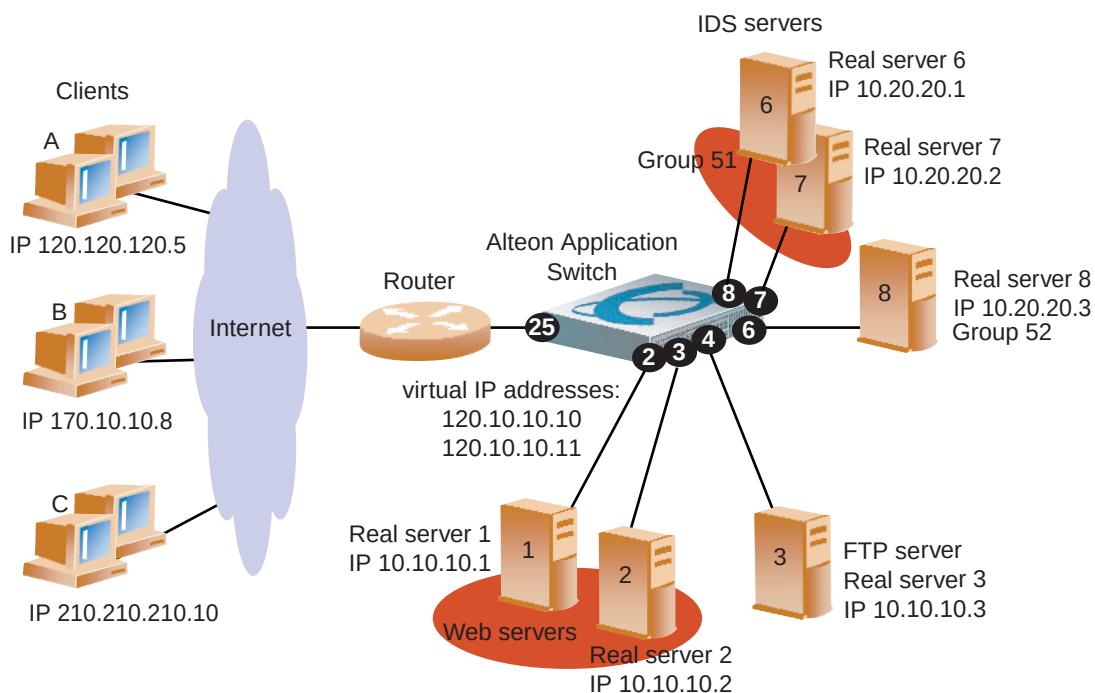


Figure 9-15 Server load balancing and IDS load balancing

When the same Alteon Application Switch is configured to load balance real servers and IDS servers as shown in [Figure 9-15](#), filter processing is not required on the client processing port (port 25). To maintain session persistency however, if you add the filter to the client port, the switch can be configured to hash on both the client IP and virtual server IP. This will ensure that both client and server traffic belonging to the same session is sent to the same IDS server. If you do not add the filter on port 25, then the switch hashes on the client IP address only.

Follow this procedure to configure the topology illustrated in [Figure 9-15](#):

1. Set up the IDS servers.

Refer to [Table 9-7 on page 191](#) for information on setting up the IDS servers. To configure the IDS servers as real servers you must consider the following:

- Connecting the IDS servers
- Choosing the health check
- Configuring the IP addresses

For more information on each of the above items, see [Step 1 on page 191](#).

2. Configure the IDS servers as real servers.

In [Figure 9-15](#) the IDS servers are setup with non-routable IP addresses. The real servers must be numbered 1-63.

```
>> # /cfg/slb/real 6/rip 10.20.20.1/ena (Specify IP address for IDS
server 6)
>> # /cfg/slb/real 7/rip 10.20.20.2/ena (Specify IP address for IDS
server 7)
>> # /cfg/slb/real 8/rip 10.20.20.3/ena (Specify IP address for IDS
server 8)
```

3. Create two IDS groups and add the IDS servers.

Define the two IDS groups 51 and 52. The IDS group numbers must be between 1 to 63.

```
>> # /cfg/slb/group 51 (Define a group)
>>Real Server Group 51# add 6 (Add IDS server 6)
>>Real Server Group 51# add 7 (Add IDS server 7)
>>Real Server Group 51# ../group 52 (Define another group)
>>Real Server Group 52# add 8 (Add IDS server 8)
```

4. Define the group metric for the IDS server groups.

IDS server load balancing supports hash and minmiss metrics only.

```
>>Real Server Group 51# metric hash           (Set the metric to hash)
>>Real Server Group 51# ../group 52           (Select the other IDS group)
>>Real Server Group 52# metric hash           (Set the metric to hash)
```

The hash metric is implemented in two ways. For more information, see [Step 4](#) on [page 192](#).

5. Define the health check for the group.

Configure ICMP health check for the group.

```
>>Real Server Group 51# health icmp           (Set the health check to ICMP)
>>Real Server Group 51# ../group 52           (Select the other IDS group)
>>Real Server Group 52# health arp           (Set the health check to ARP)
```

6. Define the group for IDS server load balancing.

```
>>Real Server Group 51# idslb ena             (Enable IDS for the IDS server group)
>>Real Server Group 51# ../group 52           (Select the other IDS group)
>>Real Server Group 52# idslb ena             (Enable IDS for the IDS server group)
```

7. Select the rport for the IDS group.

```
>> # /cfg/slb/group 51                       (Select the IDS group)
>>Real Server Group 51# idsrprt http          (Select HTTP traffic for IDS group)
>>Real Server Group 51# ../group 52           (Select the IDS group)
>>Real Server Group 52# idsrprt any(Select non-HTTP traffic for IDS
group)
```

8. Enable IDS on the client and server processing ports.

This enables frames ingressing the port to be copied to the IDS servers.

```
>># /cfg/slb/port 25/idslb ena               (Enable IDS SLB for port 25)
>>SLB port 25# ../port 2/idslb ena           (Enable IDS SLB for port 2)
>>SLB port 2# ../port 3/idslb ena           (Enable IDS SLB for port 3)
>>SLB port 3# ../port 4/idslb ena           (Enable IDS SLB for port 4)
```

In addition to enabling IDS at the port level, a filter must be configured to create a session entry for non-SLB frames ingressing the port. IDS load balancing requires a session entry to be created to store the information regarding which IDS server to send to.

9. Create an allow filter and configure the filter with the `idshash` metric.

```
>> # /cfg/slb/filt 2048           (Select the menu for Filter 2048)
>> Filter 2048# sip any          (From any source IP address)
>> Filter 2048# dip any          (To any destination IP address)
>> Filter 2048# action allow     (Allow matching traffic to pass)
>> Filter 2048# ena             (Enable the filter)
>> Filter 2048# adv/idshash both (Set the hash metric parameter)
```

The IDS hash metric is set to hash on both the source and destination IP addresses. Hashing on both source and destination IP address ensures that the returning traffic goes to the same IDS server. By default, the IDS hash metric hashes on the source IP address only.

10. Apply the filter to ports 2, 3, and 4 only.

Enable filter processing on all ports that have IDS enabled.

If you add the allow filter to the client port 25, the switch will hash on client IP and virtual server IP address for both client and server frames. This will ensure that both client and server traffic belonging to the same session is sent to the same IDS server. If you do not add the allow filter on port 25, then the switch hashes on client IP only for client frames and hashes on client IP and virtual server IP address for server frames.

```
>> # /cfg/slb/port 2             (Select the port menu)
>> SLB Port 2# add 2048          (Apply the filter to port 2)
>> SLB Port 2# filter ena       (Enable the filter)
>> SLB Port 2# ../port 3        (Select port 3)
>> SLB Port 3# add 2048          (Apply the filter to port 3)
>> SLB Port 3# filter ena       (Enable the filter)
>> SLB Port 3# ../port 4        (Select port 4)
>> SLB Port 4# add 2048          (Apply the filter to port 4)
>> SLB Port 4# filter ena       (Enable the filter)
>> SLB Port 4# ../port 25       (Select port 25)
>> SLB Port 25# add 2048        (Apply the filter to port 25)
>> SLB Port 25# filter ena      (Enable the filter)
```

11. Apply and save your changes.

```
>> SLB Port 25# apply  
>> SLB Port 25# save
```

A copy of layer 4 Web traffic from clients A, B, and C and from the real servers 1, 2, and 3 is load balanced between IDS servers 6 and 7. A copy of all non-HTTP traffic is directed to IDS server 8.

12. Configure the switch for server load balancing the real servers as described in [“Configuring Server Load Balancing” on page 144](#).

- Configure the IP interfaces on the switch
- Configure and create a group for the SLB real servers
- Configure the virtual IP address
- Configure the SLB metric
- Enable SLB

WAN Link Load Balancing

Wide Area Networking (WAN) is a telecommunications network system spread across a broad geographic area. A WAN may be privately owned or rented, but the term usually means the inclusion of public (shared user) networks, such as the telephone system. WANs can also be connected through leased lines and satellites. WANs are typically composed of powerful routers and switches that link business enterprises, universities, remote offices, and so on, around the world.

To handle the high volume of data on the Internet, some corporations are using more than one Internet Service provider (ISP) as a way to increase reliability of Internet connections. Such enterprises with more than one ISP are referred to as being *multi-homed*. In addition to reliability, a multi-homed network architecture enables enterprises to distribute load among multiple connections and to provide more optimal routing.

The WAN link load-balancing feature introduces additional resilience for networks in multi-homed environment. When users want to control which WAN link the traffic traverses, WAN link load balancing can be used to steer requests initiated within the user's network and his/her responses over the appropriate link at that moment in time.

How WAN Link Load Balancing Works

The Alteon Application Switch uses redirection filters to redirect traffic initiated from within the user's network to a group of devices that exist at the other end of the WAN link (routers, for example). These filters determine which link is the best at the time the request is generated. To ensure that the responses traverse the same link, the source IP address of the request is translated to one of the addresses that the selected ISP owns.

The design of WAN link load balancing is identical to standard redirection, except that it substitutes the source IP address of each frame with the proxy IP address of the port to which the WAN link is connected.

NOTE – The Alteon Application Switch supports a maximum of four WAN links, because you can configure four proxy IP addresses only.

The next two sections explain how WAN link load balancing works slightly different for outbound traffic versus inbound traffic.

Outbound Traffic

Outbound traffic is data from the intranet accessing a content server across the internet. In [Figure 9-16](#), outbound traffic from servers 3, 4, 5, 6, or 7 sending HTTP request to the internet will reach ports 1, 2, 3, 4, and 5 which are configured with a redirection filter for link load balancing. If the number of real servers are more than the number of ports on the Alteon Application Switch, you may connect the real servers to a layer 2 switch.

In [Figure 9-16](#) the data gets load balanced between port 25 and 26 depending on the metric of the WAN group. At ports 25 and 26, the source IP address is substituted with the proxy IP address of the port. On its return, the proxy IP address becomes the destination IP address, thus ensuring that the returning packet comes through the same WAN link. At the switch, proxy IP address is substituted back with the client IP address.

Inbound Traffic

In the inbound traffic scenario, traffic is entering the Alteon Application Switch from an external client. For example in [Figure 9-16](#), a client request `http://nortelnetworks.com` enters the switch via an ISP.

External client request can be one of the following ways:

- Accessing data from an SLB group

The client request uses the path ISP 1—Router 1—port 25 to reach its destination on server 3, 4, or 5. The virtual IP address on the switch load balances the session to the appropriate server. The returning data from the server reaches port 1 which is enabled for server processing.

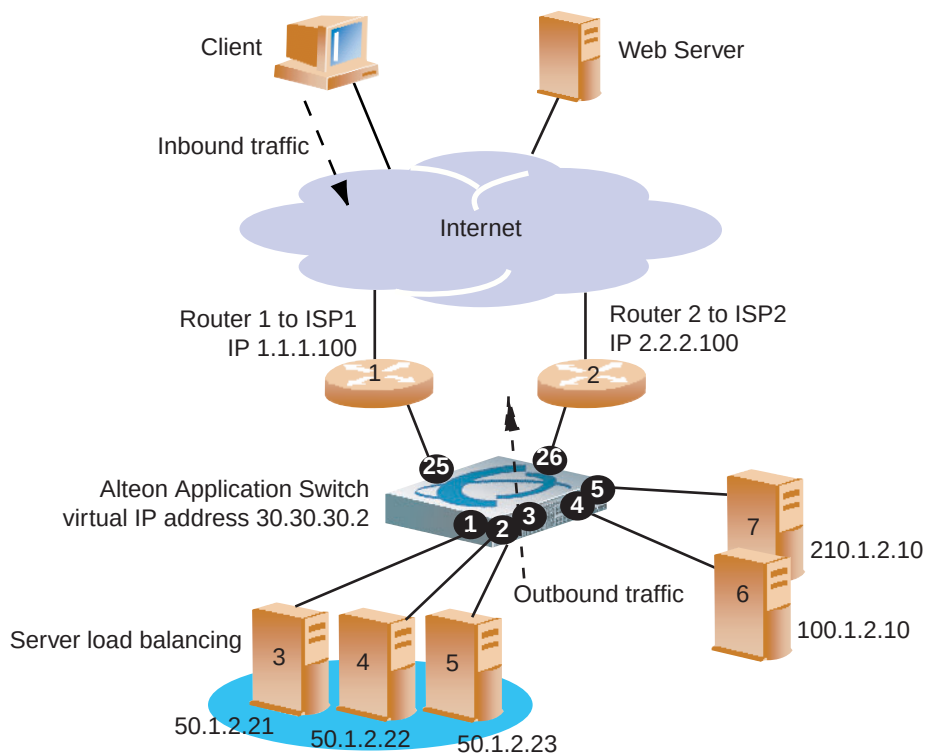


Figure 9-16 WAN Link Load Balancing

- Accessing data from a non-SLB group

The client request uses the path ISP 1—Router 1—port 25 to reach its destination on server 6 or 7. When the returning traffic from servers 6 or 7 reaches port 4 or 5, server processing is not performed, because the two servers are not in the server load balancing group.

An allow filter at ports 4 and 5 processes the data for the service configured on the server. For example, if the client sends HTTP request to server 6 or 7, then the allow filter should be configured for source port 80. Similarly, if the client sends SMTP request to server 6 or 7, then the allow filter should be configured for source port 25.

Maintaining Persistency

To maintain persistence with the session request, so that the returning session takes the same ISP path, RTS is enabled on the WAN link ports (port 25 and 26). Enabling RTS on port 25 and 26 allow the switch to associate the session with the MAC address of the WAN router. This ensures that the returning traffic takes the same ISP path.

RTS creates a reverse lookup entry on the VMA port for ingress traffic on WAN ports 25 and 26. Ingress traffic on ports 1 through 5 look at the VMA port and directs the traffic to the correct WAN port. Thus, data leaves the Alteon Application Switch from the same WAN link that was used to enter the switch, thus maintaining persistency.

Configuring WAN Link Load Balancing

Follow this procedure to configure the topology illustrated in [Figure 9-16](#):

1. Before you start configuring for WAN Link load balancing:

- Connect each WAN link to a different port on the switch.
Each WAN link requires its own port, because the switch uses the proxy IP address of the destination port when it translates the source IP address of the requests.
- Do not configure your WAN link ports into trunk groups.
- Disable NAT Cache Redirection (`cfg/slb/filt/NAT`). WAN Link load balancing and NAT cache redirection cannot be configured on the same switch. For more information, see [“IP Proxy Addresses for NAT”](#) on page 252.

2. Configure the switch for basic server load balancing.

```
>> # /cfg/slb/on
```

3. Configure IP addresses for WAN link routers.

```
>> # /cfg/slb/real 1/rip 1.1.1.100      (Define IP address for WAN link 1)
>> Real server 1# ena                  (Enable real server 1)
>> Real server 1# proxy dis            (Disable proxy)
>> # /cfg/slb/real 2/rip 2.2.2.100      (Define IP address for WAN link 2)
>> Real server 2# ena                  (Enable real server 2)
>> Real server 2# proxy dis            (Disable proxy)
```

4. Configure proxy IP addresses on ports 25 and 26.

Alteon Application Switch allows you to configure a maximum of four proxy IP addresses.

```
>> # /cfg/slb/pip                               (Select network port 25)
>> SLB pip# pip1 1.1.1.2                         (Set proxy IP address for port 25)
>> SLB pip# pip3                               (Select network port 26)
>> SLB pip# pip3 2.2.2.7                         (Set proxy IP address for port 26)
```

Use proxy IPs for port 25 and 26 for WAN link load balancing. Note that you must still configure proxy IPs for all the Gig ports on the switch. The remaining two proxy IPs are not used in the inbound or outbound traffic, so any proxy IP can be used for the other ports; however, each proxy IP address must be unique on your network.

5. Create a group to load balance the WAN link routers.

```
>> # /cfg/slb/group 100                         (Define a group)
>>Real Server Group 100# add 1                  (Add real server 1)
>>Real Server Group 100# add 2                  (Add real server 2)
```

6. Assign the response metric for the WAN link group.

```
>>Real Server Group 100# metric response        (Set the metric to response)
```

Any of the server load balancing metrics may be used, but response or bandwidth metric is recommended.

7. Configure health check for the WAN link group.

```
>>Real Server Group 100# health icmp           (Set health check to ICMP)
```

8. Define the WAN link load balancing redirection filter.

```
>> # /cfg/slb/filt 100
>> Filter 100# ena
>> Filter 100# action redir
>> Filter 100# group 100
```

9. Enable WAN link load balancing for the redirection filter.

```
>> Filter 100# adv
>> Filter 100 Advanced# linklb ena
```

10. Add link load balancing filter 100 to server ports.

>> # /cfg/slb/port 1	(Select port 1)
>> SLB Port 1# add 100	(Add filter 100 to port 1)
>> SLB Port 1# filt ena	(Enable the filter)
>> SLB Port 1# ../port 2	(Select port 2)
>> SLB Port 2# add 100	(Add filter 100 to port 2)
>> SLB Port 2# filt ena	(Enable the filter)
>> SLB Port 2# ../port 3	(Select port 3)
>> SLB Port 3# add 100	(Add filter 100 to port 3)
>> SLB Port 3# filt ena	(Enable the filter)
>> SLB Port 3# ../port 4	(Select port 4)
>> SLB Port 4# add 100	(Add filter 100 to port 4)
>> SLB Port 4# filt ena	(Enable the filter)
>> SLB Port 4# ../port 5	(Select port 5)
>> SLB Port 5# add 100	(Add filter 100 to port 5)
>> SLB Port 5# filt ena	(Enable the filter)

11. Configure IP addresses for the servers in the server load balancing group.

>> # /cfg/slb/real 3/rip 50.1.2.21/ena	(Define IP address for Real server 3)
>> # /cfg/slb/real 4/rip 50.1.2.22/ena	(Define IP address for Real server 4)
>> # /cfg/slb/real 5/rip 50.1.2.23/ena	(Define IP address for Real server 5)

12. Create a group of servers for server load balancing.

>> # /cfg/slb/group 200	(Define a group)
>>Real Server Group 200# add 3	(Add real server 3)
>>Real Server Group 200# add 4	(Add real server 4)
>>Real Server Group 200# add 5	(Add real server 5)

13. Define a virtual server and services.

>> # /cfg/slb/virt 1	(Select the virtual server)
>>Virtual Server 1# vip 30.30.30.2	(Set IP address for the virtual server)
>>Virtual Server 1# ena	(Enable virtual server)
>>Virtual Server 1# service 80	(Add the HTTP service for the virtual server)
>>Virtual Server 1 HTTP Service# group 200	(Add real server group)

14. Enable RTS for ports 25 and 26.

Enable RTS to ensure that return traffic from all servers passes through the same ISP router.

```
>> # /cfg/slb/port 25/rtb ena           (Enable rts for port 25)
>> # /cfg/slb/port 26/rtb ena           (Enable rts for port 26)
```

15. Enable client processing for ports 25 and 26 and server processing for port 1 through 5.

```
>> # /cfg/slb/port 25/client ena        (Enable client processing for port 25)
>> # /cfg/slb/port 26/client ena        (Enable client processing for port 26)
>> # /cfg/slb/port 1/server ena          (Enable server processing for port 1)
>> # /cfg/slb/port 2/server ena          (Enable server processing for port 2)
>> # /cfg/slb/port 3/server ena          (Enable server processing for port 3)
>> # /cfg/slb/port 4/server ena          (Enable server processing for port 4)
>> # /cfg/slb/port 5/server ena          (Enable server processing for port 5)
```

This enables inbound traffic to access the virtual IP address.

16. Apply and save your changes.

```
>>Real Server Group 200# apply
>>Real Server Group 200# save
```



CHAPTER 10

Filtering

This chapter provides a conceptual overview of filters and includes configuration examples showing how filters can be used for network security and Network Address Translation (NAT). The following topics are discussed in this chapter:

- **“Overview” on page 206.** This section describes the benefits and filtering criteria to allow for extensive filtering at the IP and TCP/UDP levels.
 - **“Filtering Benefits” on page 206**
 - **“Filtering Criteria” on page 206**
 - **“Stacking Filters” on page 207**
 - **“Overlapping Filters” on page 208**
 - **“The Default Filter” on page 209**
 - **“VLAN-based Filtering” on page 210**
 - **“Optimizing Filter Performance” on page 212**
 - **“Filter Logs” on page 212**
 - **“IP Address Ranges” on page 214**
 - **“Cache-Enabled versus Cache-Disabled Filters” on page 214**
- **“TCP Rate Limiting” on page 215.** This section explains how TCP rate limiting allows you to monitor the number of new TCP connections within a configurable time window.
- **“Tunable Hash for Filter Redirection” on page 220** allows you to select any hash parameter for filter redirection.
- **“Filter-based Security” on page 221.** This section provides an example of configuring filters for providing the best security.
- **“Network Address Translation” on page 227.** This section provides two examples: Internal client access to the Internet and external client access to the server.

- [“Matching TCP Flags” on page 233](#) and [“Matching ICMP Message Types” on page 238](#). Describes the ACK filter criteria which provides greater filtering flexibility and lists ICMP message types that can be filtered respectively.

Overview

Alteon Application Switches are used to deliver content efficiently and secure your servers from unauthorized intrusion, probing, and Denial-of-Service (DoS) attacks. Alteon OS includes extensive filtering capabilities at the IP and TCP/UDP levels.

Filtering Benefits

Layer 3 (IP) and Layer 4 (TCP port) filtering give the network administrator a powerful tool with the following benefits:

- Increased security for server networks
Filters can be configured to allow or deny traffic according to various IP address, protocol, and Layer 4 port criteria. You can also secure your switch from further virus attacks by allowing you to configure the switch with a list of potential offending string patterns. For more information, see [“Layer 7 Deny Filters” on page 468](#).
This gives the administrator control over the types of traffic permitted through the switch. Any filter can be optionally configured to generate system log messages for increased security visibility.
- Used to map the source or destination IP addresses and ports
Generic Network Address Translation (NAT) can be used to map the source or destination IP addresses and the ports of private network traffic to or from advertised network IP addresses and ports.

Filtering Criteria

Up to 2048 filters can be configured on the Alteon Application Switch. Descriptive names can be used to define filters. Each filter can be set to allow, deny, redirect, or translate traffic based on any combination of the following filter options:

- sip: source IP address or range (see [“IP Address Ranges” on page 214](#))
- dip: destination IP address or range (dip and dmask)
- proto: protocol number or name as shown in [Table 10-1 on page 207](#).

Table 10-1 Well-Known Protocol Types

Number	Protocol Name
1	icmp
2	igmp
6	tcp
17	udp
89	ospf
112	vrrp

- **sport**: TCP/UDP application or source port as shown in [Table 9-3 on page 148](#), or source port range (such as 31000-33000)

NOTE – The service number specified on the switch must match the service specified on the server.

- **dport**: TCP/UDP application or destination port as shown in [Table 9-3 on page 148](#), or destination port range (such as 31000-33000)
- **invert**: reverse the filter logic in order to activate the filter whenever the specified conditions are *not* met.
- Advanced filtering options such as TCP flags ([page 233](#)) or ICMP message types ([page 238](#)) are also available.

Using these filter criteria, you could create a single filter that blocks external Telnet traffic to your main server except from a trusted IP address. Another filter could warn you if FTP access is attempted from a specific IP address. Another filter could redirect all incoming e-mail traffic to a server where it can be analyzed for spam. The options are nearly endless.

Stacking Filters

Stacking filters are assigned and enabled on a per-port basis. Each filter can be used by itself or in combination with any other filter on any given switch port. The filters are numbered 1 through 2048 on Alteon Application Switches. When multiple filters are stacked together on a port, the filter’s number determines its order of precedence: the filter with the lowest number is checked first. When traffic is encountered at the switch port, if the filter matches, its configured action takes place and the rest of the filters are ignored. If the filter criteria do not match, the next filter is tried.

As long as the filters do not overlap, you can improve filter performance by making sure that the most heavily utilized filters are applied first. For example, consider a filter system where the Internet is divided according to destination IP address:

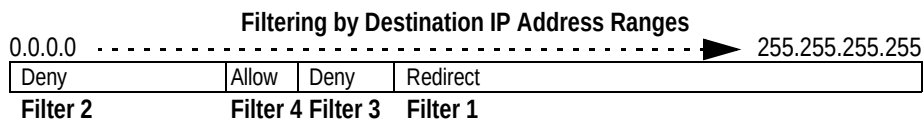


Figure 10-1 Assigning Filters According to Range of Coverage

Assuming that traffic is distributed evenly across the Internet, the largest area would be the most utilized and is assigned to Filter 1. The smallest area is assigned to Filter 4.

Overlapping Filters

Filters are permitted to overlap, although special care should be taken to ensure the proper order of precedence. When overlapping filters are present, the more specific filters (those that target fewer addresses or ports) should be applied before the generalized filters.

Example:

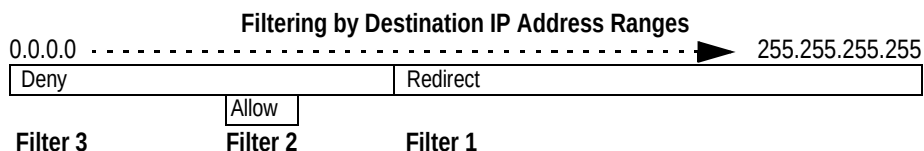


Figure 10-2 Assigning Filters to Overlapping Ranges

In this example, Filter 2 must be processed prior to Filter 3. If Filter 3 was permitted to take precedence, Filter 2 could never be triggered.

The Default Filter

Before filtering can be enabled on any given port, a default filter should be configured. This filter handles any traffic not covered by any other filter. All the criteria in the default filter must be set to the full range possible (any). For example:

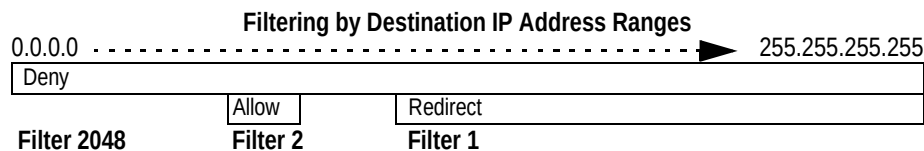


Figure 10-3 Assigning a Default Filter

In this example, the default filter is defined as Filter 2048 in order to give it the lowest order of precedence. All matching criteria in Filter 2048 are set to the any state. If no other filter acts on the traffic, Filter 2048 processes it, denying and logging unwanted traffic.

```
>> # /cfg/slb/filt 2048 (Select the default filter)
>> Filter 2048# sip any (From any source IP addresses)
>> Filter 2048# dip any (To any destination IP addresses)
>> Filter 2048# proto any (For any protocols)
>> Filter 2048# action deny (Deny matching traffic)
>> Filter 2048# name deny unwanted traffic (Provide a descriptive name for the
filter)
>> Filter 2048# ena (Enable the default filter)
>> Filter 2048# adv (Select the advanced menu)
>> Filter 2048 Advanced# log enable (Log matching traffic to syslog)
```

Default filters are recommended (but not required) when configuring filters for IP traffic control and redirection. Using default filters can increase session performance but takes some of the session binding resources. If you experience an unacceptable number of binding failures, as shown in the Server Load Balancing Maintenance Statistics (/stats/slb/maint), you may wish to remove some of the default filters.

VLAN-based Filtering

Filters are applied per switch, per port, or per VLAN. VLAN-based filtering allows a single application switch to provide differentiated services for multiple customers, groups, or departments. For example, you can define separate filters for Customers A and B on the same application switch on two different VLANs. If VLANs are assigned based on data traffic, for example, ingress traffic on VLAN 1, egress traffic on VLAN 2, and management traffic on VLAN 3, filters can be applied accordingly to the different VLANs.

In the following example shown in [Figure 10-4](#), Filter 2 is configured to allow local clients on VLAN 20 to browse the Web, and Filter 3 is configured to allow local clients on VLAN 30 to Telnet anywhere outside the local intranet. Filter 2048 is configured to deny ingress traffic from VLAN 70.

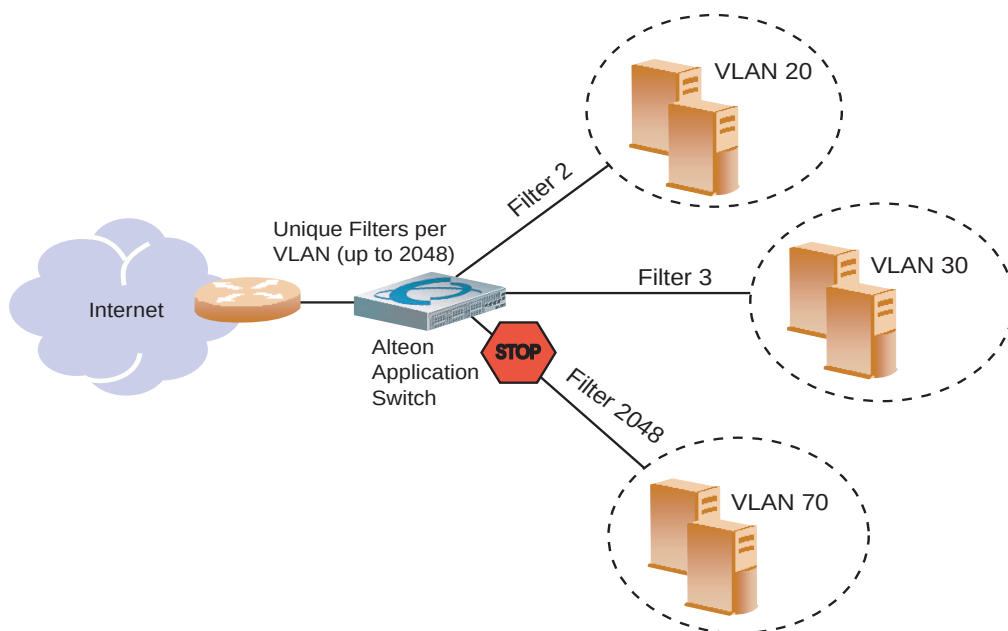


Figure 10-4 VLAN-based Filtering

Configuring VLAN-based Filtering

1. **Configure filter 2 to allow local clients to browse the Web and then assign VLAN 20 to the filter.**

The filter must recognize and allow TCP traffic from VLAN 20 to reach the local client destination IP addresses if originating from any HTTP source port:

>> # /cfg/slb/filt 2	<i>(Select the menu for Filter 2)</i>
>> Filter 2# sip any	<i>(From any source IP address)</i>
>> Filter 2# dip 205.177.15.0	<i>(To base local network dest. address)</i>
>> Filter 2# dmask 255.255.255.0	<i>(For entire subnet range)</i>
>> Filter 2# proto tcp	<i>(For TCP protocol traffic)</i>
>> Filter 2# sport http	<i>(From any source HTTP port)</i>
>> Filter 2# dport any	<i>(To any destination port)</i>
>> Filter 2# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 2# vlan 20	<i>(Assign VLAN 20 to Filter 2)</i>
>> Filter 2# ena	<i>(Enable the filter)</i>

All clients from other VLANs will be ignored.

2. **Configure filter 3 to allow local clients to Telnet anywhere outside the local intranet and then assign VLAN 30 to the filter.**

The filter must recognize and allow TCP traffic to reach the local client destination IP addresses if originating from a Telnet source port:

>> # /cfg/slb/filt 3	<i>(Select the menu for Filter 3)</i>
>> Filter 3# sip any	<i>(From any source IP address)</i>
>> Filter 3# dip 205.177.15.0	<i>(To base local network dest. address)</i>
>> Filter 3# dmask 255.255.255.0	<i>(For entire subnet range)</i>
>> Filter 3# proto tcp	<i>(For TCP protocol traffic)</i>
>> Filter 3# sport telnet	<i>(From a Telnet port)</i>
>> Filter 3# dport any	<i>(To any destination port)</i>
>> Filter 3# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 3# name allow clients to telnet	<i>(Provide a descriptive name for the filter)</i>
>> Filter 3# vlan 30	<i>(Assign VLAN 30 to Filter 3)</i>
>> Filter 3# ena	<i>(Enable the filter)</i>

3. Configure Filter 2048 to deny traffic and then assign VLAN 70 to the filter.

As a result, ingress traffic from VLAN 70 is denied entry to the switch.

>> # /cfg/slb/filt 2048	(Select the menu for Filter 2048)
>> Filter 2048# sip any	(From any source IP address)
>> Filter 2048# dip 205.177.15.0	(To base local network dest. address)
>> Filter 2048# dmask 255.255.255.0	(For entire subnet range)
>> Filter 2048# proto tcp	(For TCP protocol traffic)
>> Filter 2048# sport http	(From a Telnet port)
>> Filter 2048# dport any	(To any destination port)
>> Filter 2048# action deny	(Allow matching traffic to pass)
>> Filter 2048# vlan 70	(Assign VLAN 70 to Filter 2048)
>> Filter 2048# ena	(Enable the filter)

Optimizing Filter Performance

Filter efficiency can be increased by placing filters that are used most often near the beginning of the filtering list.

It is a recommended practice to number filters in small increments (5, 10, 15, 20, etc.) to make it easier to insert filters into the list at a later time. However, as the number of filters increases, you can improve performance by minimizing the increment between filters. For example, filters numbered 2, 4, 6, and 8 are more efficient than filters numbered 20, 40, 60, and 80. Peak processing efficiency is achieved when filters are numbered sequentially beginning with 1.

Filter Logs

To provide enhanced troubleshooting and session inspection capability, packet source and destination IP addresses are included in filter log messages. Filter log messages are generated when a Layer 3/Layer 4 filter is triggered and has logging enabled. The messages are output to the console port, system host log (syslog), and the Web-based interface message window.

Example: A network administrator has noticed a significant number of ICMP frames on one portion of the network and wants to determine the specific sources of the ICMP messages. The administrator uses the Command Line Interface (CLI) to create and apply the following filter:

```
>> # /cfg/slb/filt 15                (Select filter 15)
>> Filter 15# sip any                (From any source IP address)
>> Filter 15# dip any                (To any destination IP address)
>> Filter 15# action allow            (Allows matching traffic to pass)
>> Filter 15# name allow matching traffic (Provide a descriptive name for the
                                     filter)
>> Filter 15# proto icmp             (For the ICMP protocol)
>> Filter 15# ena                    (Enable the filter)
>> Filter 15# adv/log enable          (Log matching traffic to syslog)
>> Filter 15 Advanced# /cfg/slb/port 7 (Select a switch port to filter)
>> SLB port 7# add 15                 (Add the filter to the switch port)
>> SLB port 7# filt ena               (Enable filtering on the switch port)
>> SLB port 7# apply                  (Apply the configuration changes)
>> SLB port 7# save                   (Save the configuration changes)
```

When applied to one or more switch ports, this simple filter rule will produce log messages that show when the filter is triggered, and what the IP source and destination addresses were for the ICMP frames traversing those ports.

Example: Filter log message output is shown below, displaying the filter number, port, source IP address, and destination IP address:

```
slb: filter 15 fired on port 7, 206.118.93.110 -> 20.10.1.10
```

IP Address Ranges

You can specify a range of IP addresses for filtering both the source and/or destination IP address for traffic. When a range of IP addresses is needed, the source IP (`sip`) address or destination IP (`dip`) address defines the base IP address in the desired range. The source mask (`smask`) or destination mask (`dmask`) is the mask that is applied to produce the range.

For example, to determine if a client request’s destination IP address should be redirected to the cache servers attached to a particular switch, the destination IP address is masked (bit-wise AND) with the `dmask` and then compared to the destination IP address.

As another example, the switch could be configured with two filters so that each would handle traffic filtering for one half of the Internet. To do this, you could define the following parameters:

Table 10-2 Filtering IP Address Ranges

Filter	Internet Address Range	dip	dmask
1	0.0.0.0 - 127.255.255.255	0.0.0.0	128.0.0.0
2	128.0.0.0 - 255.255.255.255	128.0.0.0	128.0.0.0

Cache-Enabled versus Cache-Disabled Filters

To improve efficiency, by default, the Alteon Application Switch performs filter processing only on the first frame in each session. Subsequent frames in the session are assumed to match the same criteria and are automatically treated in the same way as the initial frame. These filters create a session entry in the application switch and are known as *cache-enabled*.

Some types of filtering (TCP flag and ICMP message type filtering) require each frame in the session to be filtered separately. These filters are known as *cache-disabled*.

All filters are cache-enabled by default. To change the status of a filter, use the following commands:

```
>> # /cfg/slb/filt <filter number>/adv (Select the advanced filter menu)
>> Filter 1 Advanced # cache ena|dis (Enable or disable filter caching)
```

NOTE – Cache-enabled filters should not be applied to the same ports as cache-disabled filters. Otherwise, the cache-disabled filters could potentially be bypassed for frames matching the cache-enabled criteria.

TCP Rate Limiting

Alteon OS allows you to prevent a client or a group of clients from claiming all the TCP resources on the servers. This is done by monitoring the rate of incoming TCP connection requests to a virtual IP address and limiting the client requests with a known set of IP addresses.

TCP rate limiting is similar to bandwidth management. In both features, you configure filters to limit the TCP connection requests; but in bandwidth management the limiting factor is port-based, and in TCP rate limit it is user-based. TCP rate limiting is TCP-based whereas bandwidth management is traffic-based.

The TCP rate limit is defined as the maximum number of TCP connection requests within a configured *time window*. The switch monitors the number of new TCP connections and when it exceeds the configured limit, any new TCP connection request is blocked. When this occurs, the client is said to be *held down*. The client is held down for a specified duration of time, after which new TCP connection requests from the client are allowed to pass through again.

NOTE – Both, time window and hold duration are the same for all TCP rate filters.

[Figure 10-5 on page 216](#) shows four clients configured for TCP rate limits based on source IP address. Clients 1 and 4 have the same TCP rate limit of 10 connections per second. Client 2 has a TCP rate limit of 20 connections per second. Client 3 has a TCP rate limit of 30 connections per second.

When the rate of new TCP connections from clients 1, 2, 3, and 4 reach the configured threshold, any new connection request from the client is blocked for a pre-determined amount of time. If the client's IP address and the configured filter do not match, then the default filter is applied.

In [Figure 10-5](#), the default filter 2048 configured for *Any* is applied for all other connection requests.

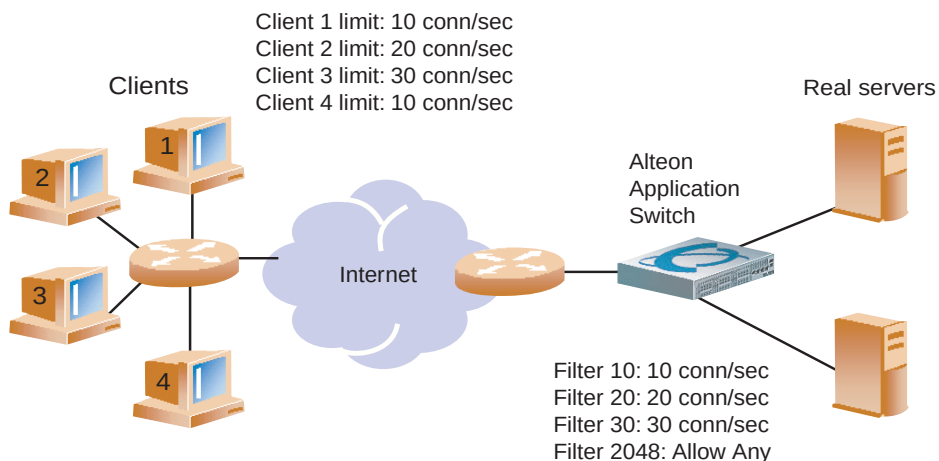


Figure 10-5 Configuring Clients with Different Rates

Configuring TCP Rate Limiting Filters

TCP rate limiting can be configured for all filter types (*allow*, *redir*, *SIP*, and *DIP*) and parameters. You can specify the source IP address and mask options in the filter configuration menu to monitor a client or a group of clients. The destination IP address and mask options are used to monitor connections to a virtual IP address or a group of virtual IP addresses.

Basic TCP Rate Limiting Filter

The following example shows how to configure TCP rate limiting for Filter 10 in [Figure 10-5](#).

1. Enable TCP rate limiting for the filter.

```
>> # /cfg/slb/filt 10/adv/tcp
>> TCP Advanced menu # tcplim ena
```

(Enable TCP rate limiting)

2. Configure maximum number of TCP connections.

```
>> TCP Advanced menu # maxcon 3
```

(Set the max. number of connections)

The `maxcon` value is specified in units of 10. The value of 3 indicates a total of 30 TCP connections.

3. Set the `timewin` parameter and calculate the total time window in seconds.

```
>> # /cfg/slb/adv/timewin 3 (Set the time window)
```

The total time window is a multiple of `fastage` (for information on `fastage`, see the Configuration chapter in the *Alteon OS Command Reference*). The total time window is calculated with the following equation:

Total Time window = `timewin` x `fastage`

If the default value for `fastage` is 1 second, then the configured total time window is 3 seconds.

NOTE – From Step 2 and 3, the TCP rate limit defined as the maximum number of connections over a specified time window is 30 TCP connections for every 3 seconds (or 10 TCP connections per second).

For a small site, 30 TCP connections per second provides a good indication if your site is being attacked. The default is 100 TCP connections per second. For larger sites, TCP rate limit greater than 2550 connection per second indicates the possibility that your switch is under attack.

4. Set the `holddur` parameter and calculate the hold down time in minutes.

```
>> # /cfg/slb/adv/holddur 2 (Set the hold duration)
```

The hold down time is a multiple of `slowage` (for information on `slowage`, see the Configuration chapter in the *Alteon OS Command Reference*). The hold down time is calculated with the following equation:

Hold down time = `holddur` x `slowage`

If `slowage` is set to the default value of 0 (2 minutes), then the configured value for hold down time is

Hold down time = 2 x 2 = 4 minutes

If a client exceeds the TCP rate limit, then the client is not allowed to make any new TCP connections for 4 minutes. The following two configuration examples illustrate how to use TCP rate limiting to limit user access based on source IP address and virtual IP address.

TCP Rate Limiting Filter Based on Source IP Address

This example shows how to define a filter that limits clients with IP address 30.30.30.x to 150 TCP connections per second. Once a user exceeds that limit, they are not allowed any new TCP connections for 10 minutes.

Configure the following on the switch:

>> # /cfg/slb/filt 100/ena	(Enable the filter)
>> Filter 100 # sip 30.30.30.0	(Specify the source IP address)
>> Filter 100 # smask 255.255.255.0	(Specify the source IP address mask)
>> Filter 100 # adv/tcp	(Select the advanced filter menu)
>> TCP advanced# tcplim en	(Enable TCP rate limiting)
>> TCP advanced# maxconn 150	(Specify the maximum connections)
>> TCP advanced# /cfg/slb/adv	(Select the Layer 4 advanced menu)
>> Layer 4 Advanced # timewin 1	(Set the time window for the session)
>> Layer 4 Advanced # holddur 5	(Set the hold duration for the session)

Fastage and slowage are set at their default values:

Fastage = 0 (1 sec) slowage = 0 (2 minutes).

Time window = timewin x fastage = 1 x 1 second = 1 second

Hold down time = holddur x slowage = 5 x 2 minutes = 10 minutes

Max rate = maxcon/time window = 150 connections/1 second = 150 connections/second

Any client with source IP address equal to 30.30.30.x is allowed to make 150 new TCP connections per second to any single destination. When the rate limit of 150 is met, the hold down time takes effect and the client is not allowed to make any new TCP connections to the same destination for 10 minutes.

TCP Rate Limiting Filter Based on Virtual Server IP Address

This example defines a filter that limits clients to 100 TCP connections per second to a specific destination (VIP 10.10.10.100). Once a client exceeds that limit, the client is not allowed to make any new TCP connection request to that destination for 40 minutes. Figure 10-6 shows how to use this feature to limit client access to a specific destination.

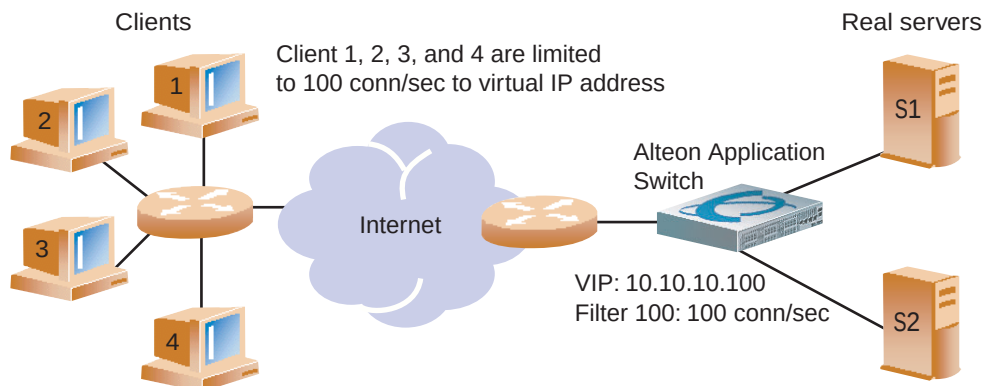


Figure 10-6 Limiting User Access to Server

Configure the following on the switch:

```
>> # /cfg/slb/filt 100/ena (Enable the filter)
>> Filter 100 # dip 10.10.10.100/dmask 255.255.255.0 (Specify the virtual server IP address)
>> Filter 100# adv/tcp (Select the advanced filter menu)
>> TCP advanced# tcplim en (Enable TCP rate limiting)
>> TCP advanced# maxconn 20 (Specify the maximum connections)
>> TCP advanced# /cfg/slb/adv (Select the Layer 4 advanced menu)
>> Layer 4 Advanced # timewin 1 (Set the time window for the session)
>> Layer 4 Advanced # holddur 5 (Set the hold duration for the session)
```

Fastage and slowage are set to 2 seconds and 8 minutes as follows:

```
/cfg/slb/adv/fastage 1 (Fastage is set to 2 seconds)
/cfg/slb/adv/slowage 2 (Slowage is set to 8 minutes)
```

time window = timewin x fastage = 1 x 2 seconds = 2 seconds

hold down time = holddur x slowage = 5 x 8 minutes = 40 minutes

max rate = maxcon/time window = 200 connections/2 seconds = 100 connections/second

All clients are limited to 100 new TCP connections/second to the server. If a client exceeds this rate, then the client is not allowed to make any new TCP connections to the server for 40 minutes.

NOTE – All SLB sessions on the switch are affected when you make changes to the `fastage` or `slowage` parameters.

Tunable Hash for Filter Redirection

Alteon OS allows you to choose a number of options when using the hash parameter for filter redirection. Hashing can be based on source IP address, destination IP address, both, or source IP address and source port. For example:

1. Configure hashing based on source IP address:

>> # /cfg/slb/filt 10/ena	(Enable the filter)
>> Filter 10 # action redir	(Specify the redir action)
>> Filter 10 # proto tcp	(Specify the protocol)
>> Filter 10 # group 1	(Specify the group of real servers)
>> Filter 10 # rport 3128	(Specify the redirection port)
>> Filter 10 # vlan any	(Specify the VLAN)
>> Filter 10 # adv	(Select the advanced filter menu)
>> TCP advanced menu # thash sip	(Select source IP address for hashing)

Hashing on the 24-bit source IP address ensures that client requests access the same cache.

2. Set the metric for the real server group to minmisses or hash.

The source IP address is passed to the real server group for either of the two metrics.

>> # /cfg/slb/group 1	(Select the group of real servers)
>> Real server group 1 # metric minmiss	(Set the metric to minmiss or hash)

NOTE – If firewall load balancing is enabled on the switch, the firewall load balancing filter which hashes on source and destination IP addresses will override the tunable hash filter.

Filter-based Security

This section provides an example of configuring filters for providing the best security. It is recommended that you configure filters to deny all traffic except for those services that you specifically wish to allow. Consider the following sample network:

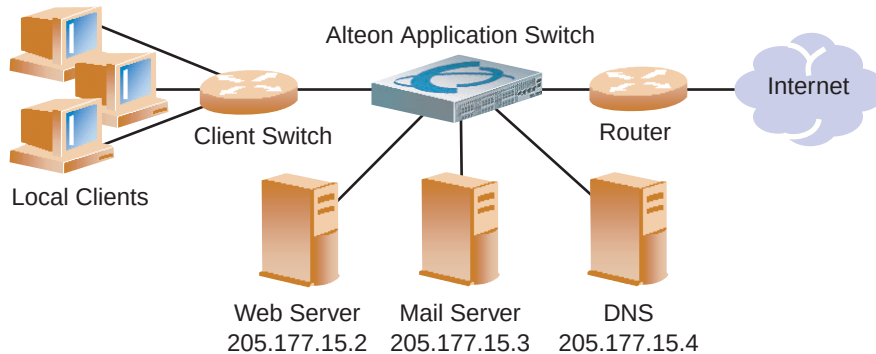


Figure 10-7 Security Topology Example

In this example, the network is made of local clients on a collector switch, a Web server, a mail server, a domain name server, and a connection to the Internet. All the local devices are on the same subnet.

In this example, the administrator wishes to install basic security filters to allow only the following traffic:

- External HTTP access to the local Web server
- External SMTP (mail) access to the local mail server
- Local clients browsing the World Wide Web
- Local clients using Telnet to access sites outside the intranet
- DNS traffic

All other traffic is denied and logged by the default filter.

NOTE – Since IP address and port information can be manipulated by external sources, filtering does not replace the necessity for a well-constructed network firewall.

Configuring a Filter-Based Security Solution

Before you begin, you must be connected to the switch CLI as the administrator.

In this example, all filters are applied only to the switch port that connects to the Internet. If intranet restrictions are required, filters can be placed on switch ports connecting to local devices.

Also, filtering is not limited to the few protocols and TCP or UDP applications shown in this example. See [Table 9-3 on page 148](#) for a list of well-known applications ports and [Table 10-1 on page 207](#) for a list of supported protocols.

1. Assign an IP address to each of the network devices.

For this example, the network devices have the following IP addresses on the same IP subnet:

Table 10-3 Web Cache Example: Real Server IP Addresses

Network Device	IP address
Local Subnet	205.177.15.0 - 205.177.15.255
Web Server	205.177.15.2
Mail Server	205.177.15.3
Domain Name Server	205.177.15.4

2. At the Application Switch, create a default filter to deny and log unwanted traffic.

The default filter is defined as Filter 2048 in order to give it the lowest order of precedence:

>> # /cfg/slb/filt 2048

>> Filter 2048# sip any

>> Filter 2048# dip any

>> Filter 2048# proto any

>> Filter 2048# action deny

>> Filter 2048# name deny unwanted traffic

>> Filter 2048# ena

>> Filter 2048# adv/log enable

(Select the default filter)

(From any source IP addresses)

(To any destination IP addresses)

(For any protocols)

(Deny matching traffic)

(Provide a descriptive name for the filter)

(Enable the default filter)

(Log matching traffic to syslog)

NOTE – Because the `proto` parameter is *not* `tcp` or `udp`, the source port (`sport`) and destination port (`dport`) values are ignored and may be excluded from the filter configuration.

3. Create a filter that will allow external HTTP requests to reach the Web server.

The filter must recognize and allow TCP traffic with the Web server's destination IP address and HTTP destination port:

```
>> Filter 2048# ../filt 1           (Select the menu for filter 1)
>> Filter 1# sip any                 (From any source IP address)
>> Filter 1# dip 205.177.15.2        (To Web server dest. IP address)
>> Filter 1# dmask 255.255.255.255  (Set mask for exact dest. address)
>> Filter 1# proto tcp               (For TCP protocol traffic)
>> Filter 1# sport any               (From any source port)
>> Filter 1# dport http              (To an HTTP destination port)
>> Filter 1# action allow            (Allow matching traffic to pass)
>> Filter 1# name allow matching traffic (Provide a descriptive name for the filter)
>> Filter 1# ena                    (Enable the filter)
```

4. Create a pair of filters to allow incoming and outgoing mail to and from the mail server.

Filter 2 allows incoming mail to reach the mail server, and Filter 3 allows outgoing mail to reach the Internet:

```
>> Filter 1# ../filt 2           (Select the menu for filter 2)
>> Filter 2# sip any             (From any source IP address)
>> Filter 2# dip 205.177.15.3    (To mail server dest. IP address)
>> Filter 2# dmask 255.255.255.255 (Set mask for exact dest. address)
>> Filter 2# proto tcp           (For TCP protocol traffic)
>> Filter 2# sport any           (From any source port)
>> Filter 2# dport smtp          (To a SMTP destination port)
>> Filter 2# action allow        (Allow matching traffic to pass)
>> Filter 2# ena                 (Enable the filter)
>> Filter 2# ../filt 3           (Select the menu for filter 3)
>> Filter 3# sip 205.177.15.3    (From mail server source IP address)
>> Filter 3# smask 255.255.255.255 (Set mask for exact source address)
>> Filter 3# dip any             (To any destination IP address)
>> Filter 3# proto tcp           (For TCP protocol traffic)
>> Filter 3# sport smtp          (From a SMTP port)
>> Filter 3# dport any           (To any destination port)
>> Filter 3# action allow        (Allow matching traffic to pass)
>> Filter 3# ena                 (Enable the filter)
```

5. Create a filter that will allow local clients to browse the Web.

The filter must recognize and allow TCP traffic to reach the local client destination IP addresses if traffic originates from any HTTP source port:

```
>> Filter 3# ../filt 4                (Select the menu for Filter 4)
>> Filter 4# sip any                  (From any source IP address)
>> Filter 4# dip 205.177.15.0         (To base local network dest. address)
>> Filter 4# dmask 255.255.255.0     (For entire subnet range)
>> Filter 4# proto tcp                (For TCP protocol traffic)
>> Filter 4# sport http               (From any source HTTP port)
>> Filter 4# dport any                (To any destination port)
>> Filter 4# action allow             (Allow matching traffic to pass)
>> Filter 4# name allow clients Web browse (Provide a descriptive name for the
                                         filter)
>> Filter 4# ena                     (Enable the filter)
```

6. Create a filter that will allow local clients to Telnet anywhere outside the local intranet.

The filter must recognize and allow TCP traffic to reach the local client destination IP addresses if originating from a Telnet source port:

```
>> Filter 4# ../filt 5                (Select the menu for Filter 5)
>> Filter 5# sip any                  (From any source IP address)
>> Filter 5# dip 205.177.15.0         (To base local network dest. address)
>> Filter 5# dmask 255.255.255.0     (For entire subnet range)
>> Filter 5# proto tcp                (For TCP protocol traffic)
>> Filter 5# sport telnet             (From a Telnet port)
>> Filter 5# dport any                (To any destination port)
>> Filter 5# action allow             (Allow matching traffic to pass)
>> Filter 5# ena                     (Enable the filter)
```

7. Create a series of filters to allow Domain Name System (DNS) traffic.

DNS traffic requires four filters; one pair is needed for UDP traffic (incoming and outgoing) and another pair for TCP traffic (incoming and outgoing).

For UDP:

>> Filter 5# ../filt 6	<i>(Select the menu for Filter 6)</i>
>> Filter 6# sip any	<i>(From any source IP address)</i>
>> Filter 6# dip 205.177.15.4	<i>(To local DNS Server)</i>
>> Filter 6# dmask 255.255.255.255	<i>(Set mask for exact dest. address)</i>
>> Filter 6# proto udp	<i>(For UDP protocol traffic)</i>
>> Filter 6# sport any	<i>(From any source port)</i>
>> Filter 6# dport domain	<i>(To any DNS destination port)</i>
>> Filter 6# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 6# ena	<i>(Enable the filter)</i>
>> Filter 6# ../filt 7	<i>(Select the menu for Filter 7)</i>
>> Filter 7# sip 205.177.15.4	<i>(From local DNS Server)</i>
>> Filter 7# smask 255.255.255.255	<i>(Set mask for exact source address)</i>
>> Filter 7# dip any	<i>(To any destination IP address)</i>
>> Filter 7# proto udp	<i>(For UDP protocol traffic)</i>
>> Filter 7# sport domain	<i>(From a DNS source port)</i>
>> Filter 7# dport any	<i>(To any destination port)</i>
>> Filter 7# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 7# ena	<i>(Enable the filter)</i>

Similarly, for TCP:

>> Filter 7# ../filt 8	<i>(Select the menu for Filter 8)</i>
>> Filter 8# sip any	<i>(From any source IP address)</i>
>> Filter 8# dip 205.177.15.4	<i>(To local DNS Server)</i>
>> Filter 8# dmask 255.255.255.255	<i>(Set mask for exact dest. address)</i>
>> Filter 8# proto tcp	<i>(For TCP protocol traffic)</i>
>> Filter 8# sport any	<i>(From any source port)</i>
>> Filter 8# dport domain	<i>(To any DNS destination port)</i>
>> Filter 8# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 8# ena	<i>(Enable the filter)</i>
>> Filter 8# ../filt 9	<i>(Select the menu for Filter 9)</i>
>> Filter 9# sip 205.177.15.4	<i>(From local DNS Server)</i>
>> Filter 9# smask 255.255.255.255	<i>(Set mask for exact source address)</i>
>> Filter 9# dip any	<i>(To any destination IP address)</i>
>> Filter 9# proto tcp	<i>(For TCP protocol traffic)</i>
>> Filter 9# sport domain	<i>(From a DNS source port)</i>
>> Filter 9# dport any	<i>(To any destination port)</i>
>> Filter 9# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 9# ena	<i>(Enable the filter)</i>

8. Assign the filters to the switch port that connects to the Internet.

>> Filter 9# ../port 5	<i>(Select the SLB port 5 to the Internet)</i>
>> SLB Port 5# add 1-9	<i>(Add filters 1-9 to port 5)</i>
>> SLB Port 5# add 2048	<i>(Add the default filter to port 5)</i>
>> SLB Port 5# filt enable	<i>(Enable filtering for port 5)</i>

Alteon OS allows you to add and remove a contiguous block of filters with a single command.

9. Apply and verify the configuration.

>> SLB Port 5# apply	<i>(Make your changes active)</i>
>> SLB Port 5# cur	<i>(View current settings)</i>

Examine the resulting information. If any settings are incorrect, make appropriate changes.

10. Save your new configuration changes.

>> SLB Port 5# save	<i>(Save for restore after reboot)</i>

11. Check the server load balancing information.

>> SLB Port 5# /info/slb/dump	<i>(View SLB information)</i>

Check that all SLB parameters are working according to expectation. If necessary, make any appropriate configuration changes and then check the information again.

NOTE – Changes to filters on a given port do not take effect until the port's session information is updated (every two minutes or so). To make filter changes take effect immediately, clear the session binding table for the port (see the `/oper/slb/clear` command in the *Alteon OS Command Reference*).

Network Address Translation

Network Address Translation (NAT) is an Internet standard that enables an Alteon Application Switch to use one set of IP addresses for internal traffic and a second set of addresses for external traffic. Alteon Application Switches use filters to implement NAT.

NAT serves two main purposes:

- Provides a type of firewall by hiding internal IP addresses and increases network security.
- Enables a company to use more internal IP addresses. Since they're used internally only, there's no possibility of conflict with public IP addresses used by other companies and organizations.

In the following NAT examples, a company has configured its internal network with *private* IP addresses. A private network is one that is isolated from the global Internet and is, therefore, free from the usual restrictions requiring the use of registered, globally unique IP addresses.

With NAT, private networks are not required to remain isolated. NAT capabilities within the switch allow internal, private network IP addresses to be translated to valid, publicly advertised IP addresses and back again. NAT can be configured in one of the following two ways:

- Static NAT provides a method for direct mapping of one predefined IP address (such as a publicly available IP address) to another (such as a private IP address)
- Dynamic NAT provides a method for mapping multiple IP addresses (such as a group of internal clients) to a single IP address (to conserve publicly advertised IP addresses)

Static NAT

The static NAT (non-proxy) example requires two filters: one for the external client-side switch port, and one for the internal, server-side switch port. The client-side filter translates incoming requests for the publicly advertised server IP address to the server's internal private network address. The filter for the server-side switch port reverses the process, translating the server's private address information to a valid public address.

In this example, clients on the Internet require access to servers on the private network:

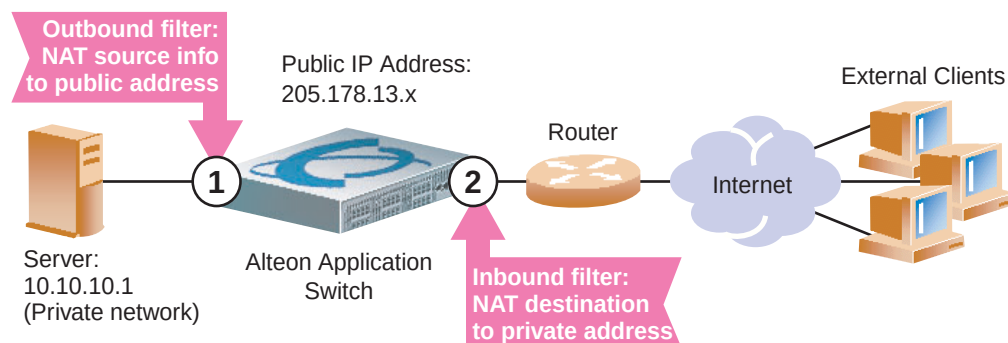


Figure 10-8 Static Network Address Translation

Configuring Static NAT

>> # /cfg/slb/filt 10	(Select the menu for outbound filter)
>> Filter 10# action nat	(Perform NAT on matching traffic)
>> Filter 10# nat source	(Translate source information)
>> Filter 10# sip 10.10.10.0	(From the clients private IP address)
>> Filter 10# smask 255.255.255.0	(For the entire private subnet range)
>> Filter 10# dip 205.178.13.0	(To the public network address)
>> Filter 10# dmask 255.255.255.0	(For the same subnet range)
>> Filter 10# ena	(Enable the filter)
>> Filter 10# adv/proxy disable	(Override any proxy IP settings)
>> Filter 10 Advanced# /cfg/slb/filt 11	(Select the menu for inbound filter)
>> Filter 11# action nat	(Use the same settings as outbound)
>> Filter 11# nat dest	(Reverse the translation direction)
>> Filter 11# sip 10.10.10.0	(Use the same settings as outbound)
>> Filter 11# smask 255.255.255.0	(Use the same settings as outbound)
>> Filter 11# dip 205.178.13.0	(Use the same settings as outbound)
>> Filter 11# dmask 255.255.255.0	(Use the same settings as outbound)
>> Filter 11# ena	(Enable the filter)
>> Filter 11# adv/proxy disable	(Override any proxy IP settings)
>> Filter 11 Advanced# /cfg/slb/port 1	(Select server-side port)
>> SLB port 1# add 10	(Add the outbound filter)
>> SLB port 1# filt enable	(Enable filtering on port 1)
>> SLB port 1# ../port 2	(Select the client-side port)
>> SLB port 2# add 11	(Add the inbound filter)
>> SLB port 2# filt enable	(Enable filtering on port 2)
>> SLB port 2# apply	(Apply configuration changes)
>> SLB port 2# save	(Save configuration changes)

Note the following important points about this configuration:

- Within each filter, the `smask` and `dmask` values are identical.
- All parameters for both filters are identical except for the NAT direction. For Filter 10, `nat source` is used. For Filter 11, `nat dest` is used.
- Filters for static (non-proxy) NAT should take precedence over dynamic NAT filters (following example). Static filters should be given lower filter numbers.

Dynamic NAT

Dynamic NAT is a many-to-one solution: multiple clients on the private subnet take advantage of a single external IP address, thus conserving valid IP addresses. In this example, clients on the internal private network require TCP/UDP access to the Internet:

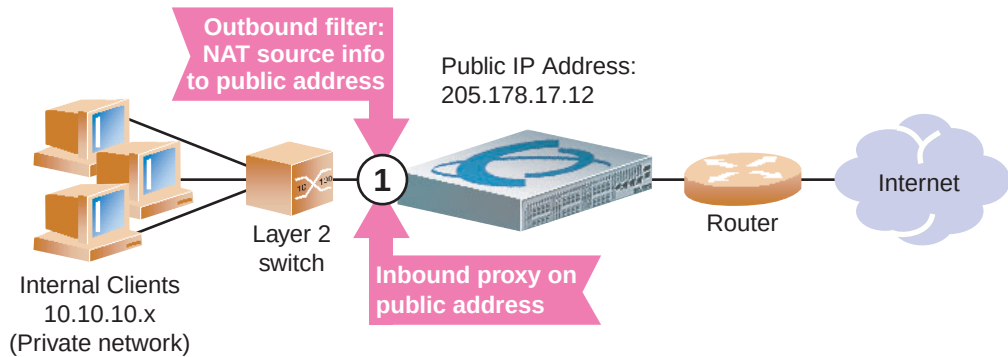


Figure 10-9 Dynamic Network Address Translation

You may directly connect the clients to the application switch if the total number of clients is less than or equal to the switch ports.

NOTE – Dynamic NAT can also be used to support ICMP traffic for PING.

This example requires a NAT filter to be configured on the switch port that is connected to the internal clients. When the NAT filter is triggered by outbound client traffic, the internal private IP address information on the outbound packets is translated to a valid, publicly advertised IP address on the switch. In addition, the public IP address must be configured as a proxy IP address on the switch port that is connected to the internal clients. The proxy performs the reverse translation, restoring the private network addresses on inbound packets.

Configuring Dynamic NAT

>> # /cfg/slb/filt 14	(Select the menu for client filter)
>> Filter 14# invert ena	(Invert the filter logic)
>> Filter 14# dip 10.10.10.0	(If the destination is not private)
>> Filter 14# dmask 255.255.255.0	(For the entire private subnet range)
>> Filter 14# sip any	(From any source IP address)
>> Filter 14# action nat	(Perform NAT on matching traffic)
>> Filter 14# nat source	(Translate source information)
>> Filter 14# ena	(Enable the filter)
>> Filter 14# adv/proxy enable	(Allow proxy IP translation)
>> Filter 14 Advanced# cfg/slb/pip	(Select the proxy IP address menu)
>> Proxy IP Address# pip1 205.178.17.12	(Set proxy IP address)
>> Proxy IP Address# ../port 1	(Select SLB port 1)
>> SLB port 1# add 14	(Add the filter to port 1)
>> SLB port 1# filt enable	(Enable filtering on port 1)
>> SLB port 1# proxy ena	(Enable proxies on this port)
>> SLB port 1# apply	(Apply configuration changes)
>> SLB port 1# save	(Save configuration changes)

For more information on proxy IP address, see [“Proxy IP Addresses” on page 156](#).

NOTE – The `invert` option in this example filter makes this specific configuration easier but is not a requirement for dynamic NAT.

NOTE – Dynamic NAT solutions apply only to TCP/UDP traffic. Also, filters for dynamic NAT should be given a higher numbers than any static NAT filters (see [“Static NAT” on page 227](#)).

FTP Client NAT

Alteon Application Switches provide NAT services to many clients with private IP addresses. In Alteon OS, an FTP enhancement provides the capability to perform true FTP NAT for dynamic NAT.

Because of the way FTP works in active mode, a client sends information on the control channel, information that reveals their private IP address, out to the Internet. However, the switch filter only performs NAT translation on the TCP/IP header portion of the frame, preventing a client with a private IP address from doing active FTP.

The switch can monitor the control channel and replace the client's private IP address with a proxy IP address defined on the switch. When a client in active FTP mode sends a `port` command to a remote FTP server, the switch will look into the data part of the frame and modify the `port` command as follows:

- The real server (client) IP address will be replaced by a public proxy IP address. If VMA is enabled, a pool (1-8) of proxy IP addresses is used instead of a single one.
- The real server (client) port will be replaced with a proxy port.

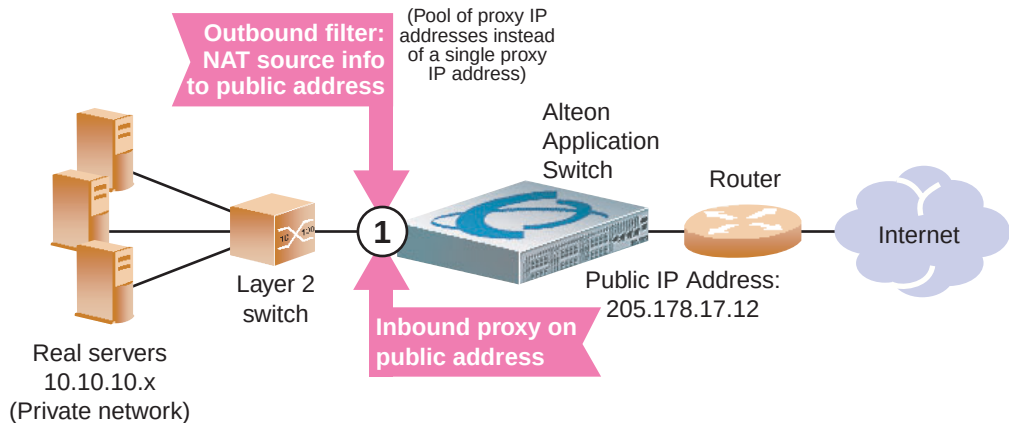


Figure 10-10 Active FTP for Dynamic NAT

You may directly connect the real servers to the application switch if the total number of servers is less than or equal to the switch ports.

Configuring Active FTP Client NAT

NOTE – The passive mode does not need this feature.

1. Make sure that a proxy IP address is enabled on the filter port.
2. Make sure that a source NAT filter is set up for the port.:

>> # /cfg/slb/filt 14	(Select the menu for client filter)
>> Filter 14# invert ena	(Invert the filter logic)
>> Filter 14# dip 10.10.10.0	(If the destination is not private)
>> Filter 14# dmask 255.255.255.0	(For the entire private subnet range)
>> Filter 14# sip any	(From any source IP address)
>> Filter 14# action nat	(Perform NAT on matching traffic)
>> Filter 14# nat source	(Translate source information)
>> Filter 14# ena	(Enable the filter)
>> Filter 14# adv/proxy enable	(Allow proxy IP translation)
>> Filter 14 Advanced# cfg/slb/pip	(Select the proxy IP address menu)
>> Proxy IP Address# pip1 205.178.17.12	(Set proxy IP address)
>> Proxy IP Address# ../port 1	(Select SLB port 1)
>> SLB port 1# add 14	(Add the filter to port 1)
>> SLB port 1# filt enable	(Enable filtering on port 1)
>> SLB port 1# proxy ena	(Enable proxies on this port)
>> SLB port 1# apply	(Apply configuration changes)
>> SLB port 1# save	(Save configuration changes)

For more information on proxy IP address, see [“Proxy IP Addresses” on page 156](#).

3. Enable active FTP NAT using the following command:

>> # /cfg/slb/filt <filter number>/adv/ftpa ena

4. Apply and save the switch configuration.

Matching TCP Flags

Alteon OS supports packet filtering based on any of the following TCP flags.

Table 10-4 TCP Flags

Flag	Description
URG	Urgent
ACK	Acknowledgement
PSH	Push
RST	Reset
SYN	Synchronize
FIN	Finish

Any filter may be set to match against more than one TCP flag at the same time. If there is more than one flag enabled, the flags are applied with a logical AND operator. For example, by setting the switch to filter SYN and ACK, the switch filters all SYN-ACK frames.

NOTE – TCP flag filters must be cache-disabled. Exercise caution when applying cache-enabled and cache-disabled filters to the same switch port. For more information, see “Cache-Enabled versus Cache-Disabled Filters” on page 214.

Configuring the TCP Flag Filter

NOTE – By default, all TCP filter options are disabled. TCP flags will *not* be inspected unless one or more TCP options are enabled.

Consider the following network:

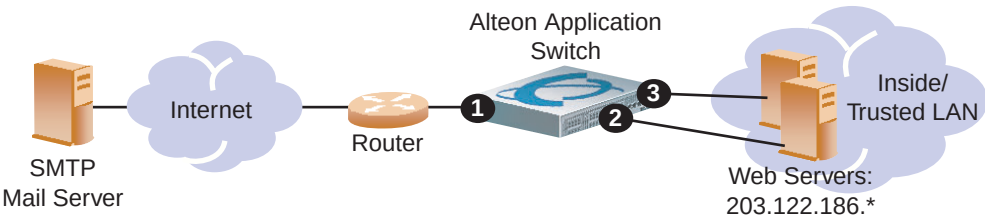


Figure 10-11 TCP ACK Matching Network

In this network, the Web servers inside the LAN must be able to transfer mail to any SMTP-based mail server out on the Internet. At the same time, you want to prevent access to the LAN from the Internet, except for HTTP.

SMTP traffic uses well-known TCP Port 25. The Web servers will originate TCP sessions to the SMTP server using TCP destination Port 25, and the SMTP server will acknowledge each TCP session and data transfer using TCP source Port 25.

Creating a filter with the ACK flag closes one potential security hole. Without the filter, the switch would permit a TCP SYN connection request to reach any listening TCP destination port on the Web servers inside the LAN, as long as it originated from TCP source Port 25. The server would listen to the TCP SYN, allocate buffer space for the connection, and reply to the connect request. In some SYN attack scenarios, this could cause the server's buffer space to fill, crashing the server or at least making it unavailable.

A filter with the ACK flag enabled prevents external devices from beginning a TCP connection (with a TCP SYN) from TCP source Port 25. The switch drops any frames that have the ACK flag turned off.

The following filters are required:

1. **An allow filter for TCP traffic from LAN that allows the Web servers to pass SMTP requests to the Internet.**

>> # /cfg/slb/filt 10	<i>(Select a filter for trusted SMTP requests)</i>
>> Filter 10# sip 203.122.186.0	<i>(From the Web servers' source IP address)</i>
>> Filter 10# smask 255.255.255.0	<i>(For the entire subnet range)</i>
>> Filter 10# sport any	<i>(From any source port)</i>
>> Filter 10# proto tcp	<i>(For TCP traffic)</i>
>> Filter 10# dip any	<i>(To any destination IP address)</i>
>> Filter 10# dport smtp	<i>(To well-known destination SMTP port)</i>
>> Filter 10# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 10# ena	<i>(Enable the filter)</i>

2. A filter that allows SMTP traffic from the Internet to pass through the switch *only* if the destination is one of the Web servers, and the frame is an acknowledgment (SYN-ACK) of a TCP session.

>> Filter 10# ../filt 15	<i>(Select a filter for Internet SMTP ACKs)</i>
>> Filter 15# sip any	<i>(From any source IP address)</i>
>> Filter 15# sport smtp	<i>(From well-known source SMTP port)</i>
>> Filter 15# proto tcp	<i>(For TCP traffic)</i>
>> Filter 15# dip 203.122.186.0	<i>(To the Web servers' IP address)</i>
>> Filter 15# dmask 255.255.255.0	<i>(To the entire subnet range)</i>
>> Filter 15# dport any	<i>(To any destination port)</i>
>> Filter 15# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 15# ena	<i>(Enable the filter)</i>
>> Filter 15# adv/tcp	<i>(Select the advanced TCP menu)</i>
>> Filter 15 Advanced# ack ena	<i>(Match acknowledgments only)</i>
>> Filter 15 Advanced# syn ena	<i>(Match acknowledgments only)</i>

3. A filter that allows SMTP traffic from the Internet to pass through the switch *only* if the destination is one of the Web servers, and the frame is an acknowledgment (ACK-PSH) of a TCP session.

>> Filter 15# ../filt 16	<i>(Select a filter for Internet SMTP ACKs)</i>
>> Filter 16# sip any	<i>(From any source IP address)</i>
>> Filter 16# sport smtp	<i>(From well-known source SMTP port)</i>
>> Filter 16# proto tcp	<i>(For TCP traffic)</i>
>> Filter 16# dip 203.122.186.0	<i>(To the Web servers' IP address)</i>
>> Filter 16# dmask 255.255.255.0	<i>(To the entire subnet range)</i>
>> Filter 16# dport any	<i>(To any destination port)</i>
>> Filter 16# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 16# ena	<i>(Enable the filter)</i>
>> Filter 16# adv/tcp	<i>(Select the advanced TCP menu)</i>
>> Filter 16 Advanced# ack ena	<i>(Match acknowledgments only)</i>
>> Filter 16 Advanced# psh ena	<i>(Match acknowledgments only)</i>

4. A filter that allows trusted HTTP traffic from the Internet to pass through the switch to the Web servers.

```
>> Filter 16 Advanced# /cfg/slb/filt 17 (Select a filter for incoming HTTP traffic)
>> Filter 17# sip any (From any source IP address)
>> Filter 17# sport http (From well-known source HTTP port)
>> Filter 17# proto tcp (For TCP traffic)
>> Filter 17# dip 203.122.186.0 (To the Web servers' IP address)
>> Filter 17# dmask 255.255.255.0 (To the entire subnet range)
>> Filter 17# dport http (To well-known destination HTTP port)
>> Filter 17# action allow (Allow matching traffic to pass)
>> Filter 17# ena (Enable the filter)
```

5. A filter that allows HTTP responses from the Web servers to pass through the switch to the Internet.

```
>> Filter 17# ../filt 18 (Select a filter for outgoing HTTP traffic)
>> Filter 18# sip 203.122.186.0 (From the Web servers' source IP address)
>> Filter 18# smask 255.255.255.0 (From the entire subnet range)
>> Filter 18# sport http (From well-known source HTTP port)
>> Filter 18# proto tcp (For TCP traffic)
>> Filter 18# dip any (To any destination IP address)
>> Filter 18# dport http (To well-known destination HTTP port)
>> Filter 18# action allow (Allow matching traffic to pass)
>> Filter 18# ena (Enable the filter)
```

6. A default filter is required to deny all other traffic.

```
>> Filter 18# ../filt 2048 (Select a default filter)
>> Filter 2048# sip any (From any source IP address)
>> Filter 2048# dip any (To any destination IP address)
>> Filter 2048# action deny (Block matching traffic)
>> Filter 2048# name deny matching traffic (Provide a descriptive name for the filter)
>> Filter 2048# ena (Enable the filter)
```

7. Apply the filters to the appropriate switch ports.

>> Filter 2048# ../port 1	<i>(Select the Internet-side port)</i>
>> SLB port 1# add 15	<i>(Add the SMTP ACK filter to the port)</i>
>> SLB port 1# add 16	<i>(Add the incoming HTTP filter)</i>
>> SLB port 1# add 17	<i>(Add the incoming HTTP filter)</i>
>> SLB port 1# add 2048	<i>(Add the default filter to the port)</i>
>> SLB port 1# filt ena	<i>(Enable filtering on the port)</i>
>> SLB port 1# ../port 2	<i>(Select the first Web server port)</i>
>> SLB port 2# add 10	<i>(Add the outgoing SMTP filter to the port)</i>
>> SLB port 2# add 18	<i>(Add the outgoing HTTP filter to the port)</i>
>> SLB port 2# add 2048	<i>(Add the default filter to the port)</i>
>> SLB port 2# filt ena	<i>(Enable filtering on the port)</i>
>> SLB port 2# ../port 3	<i>(Select the other Web server port)</i>
>> SLB port 3# add 10	<i>(Add the outgoing SMTP filter to the port)</i>
>> SLB port 3# add 18	<i>(Add the outgoing HTTP filter to the port)</i>
>> SLB port 3# add 2048	<i>(Add the default filter to the port)</i>
>> SLB port 3# filt ena	<i>(Enable filtering on the port)</i>
>> SLB port 3# apply	<i>(Apply the configuration changes)</i>
>> SLB port 3# save	<i>(Save the configuration changes)</i>

Matching ICMP Message Types

Internet Control Message Protocol (ICMP) is used for reporting TCP/IP processing errors. There are numerous types of ICMP messages, as shown in [Table 10-5](#). Although ICMP packets can be filtered using the `proto icmp` option, by default, the application switch ignores the ICMP message type when matching a packet to a filter. To perform filtering based on specific ICMP message types, ICMP message type filtering must be enabled.

Alteon OS software supports filtering on the following ICMP message types:

Table 10-5 ICMP Message Types

Type #	Message Type	Description
0	echorep	ICMP echo reply
3	destun	ICMP destination unreachable
4	quench	ICMP source quench
5	redir	ICMP redirect
8	echoreq	ICMP echo request
9	rtradv	ICMP router advertisement
10	rtrsol	ICMP router solicitation
11	timex	ICMP time exceeded
12	param	ICMP parameter problem
13	timereq	ICMP timestamp request
14	timerep	ICMP timestamp reply
15	inforeq	ICMP information request
16	inforep	ICMP information reply
17	maskreq	ICMP address mask request
18	maskrep	ICMP address mask reply

The command to enable or disable ICMP message type filtering is entered from the Advanced Filtering menu as follows:

```
>> # /cfg/slb/filt <filter number>/adv  
>> Filter 1 Advanced# icmp <message type|number|any|list>
```

For any given filter, only one ICMP message type can be set at any one time. The `any` option disables ICMP message type filtering. The `list` option displays a list of the available ICMP message types that can be entered.

NOTE – ICMP message type filters must be cache-disabled. Exercise caution when applying cache-enabled and cache-disabled filters to the same switch port. For more information, see [“Cache-Enabled versus Cache-Disabled Filters”](#) on page 214.



CHAPTER 11

Application Redirection

Application Redirection improves network bandwidth and provides unique network solutions. Filters can be created to redirect traffic to cache and application servers improving speed of access to repeated client access to common Web or application content and free valuable network bandwidth.

The following topics are discussed in this chapter:

- **“Overview” on page 242.** Application redirection helps reduce the traffic congestion during peak loads by accessing locally cached information. This section also discusses how performance is improved by balancing cached requests across multiple servers.
- **“Cache Redirection Configuration Example” on page 244.** This section provides a step-by-step procedure on how to intercept all Internet bound HTTP requests (on default TCP port 80) and redirect them to the cache servers.
- **“RTSP Cache Redirection” on page 249.** This section explains how to configure the switch to redirect data (multimedia presentations) to the cache servers, and how to balance the load among the cache servers.
- **“IP Proxy Addresses for NAT” on page 252.** This section discusses the benefits of transparent proxies when used with application redirection.
- **“Excluding Noncacheable Sites” on page 254.** This section describes how to filter out applications that prevent real-time session information from being redirected to cache servers.

NOTE – To access Application Redirection functionality, the optional Layer 4 software must be enabled on the application switch (see “Filtering and Layer 4” in the *Alteon OS Command Reference*).

Overview

Most of the information downloaded from the Internet is not unique, as clients will often access the Web page many times for additional information or to explore other links. Duplicate information also gets requested as the components that make up Internet data at a particular Web site (pictures, buttons, frames, text, and so on) are reloaded from page to page. When you consider this scenario in the context of many clients, it becomes apparent that redundant requests can consume a considerable amount of your available bandwidth to the Internet.

Application redirection can help reduce the traffic congestion during peak loads. When Application redirection filters are properly configured for the Alteon OS-powered switch, outbound client requests for Internet data are intercepted and redirected to a group of application or cache servers on your network. The servers duplicate and store inbound Internet data that has been requested by your clients. If the servers recognize a client's outbound request as one that can be filled with cached information, the servers supply the information rather than send the request across the Internet.

In addition to increasing the efficiency of your network, accessing locally cached information can be much faster than requesting the same information across the Internet.

Cache Redirection Environment

Consider a network where client HTTP requests begin to regularly overload the Internet router.

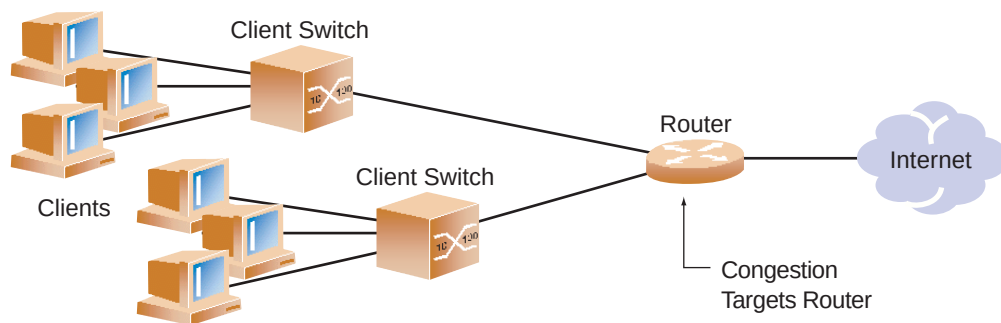


Figure 11-1 Traditional Network Without Cache Redirection

The network needs a solution that addresses the following key concerns:

- The solution must be readily scalable
- The administrator should not need to reconfigure all the clients' browsers to use proxy servers.

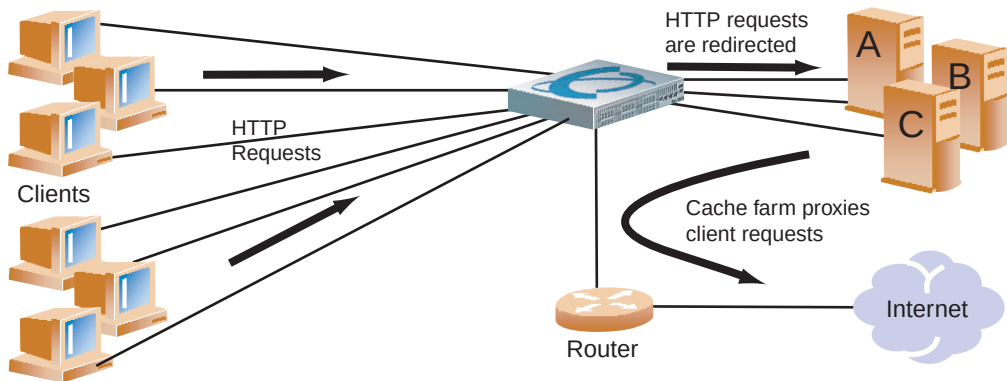


Figure 11-2 Network with Cache Redirection

If you have more clients than application switch ports, then connect the clients to a layer 2 switch.

Adding an Alteon Application Switch with optional Layer 4 software addresses these issues:

- Cache servers can be added or removed dynamically without interrupting services.
- Performance is improved by balancing the cached request load across multiple servers. More servers can be added at any time to increase processing power.
- The proxy is transparent to the client.
- Frames that are not associated with HTTP requests are normally passed to the router.

Additional Application Redirection Options

Application redirection can be used in combination with other Layer 4 options, such as load balancing metrics, health checks, real server group backups, and more. See [“Additional Server Load Balancing Options” on page 148](#) for details.

Cache Redirection Configuration Example

The following is required prior to configuration:

- You must connect to the application switch Command Line Interface (CLI) as the administrator.
- Layer 4 (SLB) software must be enabled.

NOTE – For details about the procedures above, and about any of the menu commands described in this example, see the *Alteon OS Command Reference*.

In this example, an Alteon Application Switch is placed between the clients and the border gateway to the Internet. The application switch will be configured to intercept all Internet bound HTTP requests (on default TCP port 80), and redirect them to the cache servers. The application switch will distribute HTTP requests equally to the cache servers based on the destination IP address of the requests. If the cache servers do not have the requested information, then the cache servers behave like the client and forwards the request out to the internet.

Also, filters are not limited to the few protocols and TCP or UDP applications shown in this example. See [Table 9-3 on page 148](#) for a list of well-known applications ports and [Table 10-1 on page 207](#) for a list of supported protocols.

1. Assign an IP address to each of the cache servers.

Similar to server load balancing, the cache real servers are assigned an IP address and placed into a real server group. The real servers must be in the same VLAN and must have an IP route to the application switch that will perform the cache redirection. In addition, the path from the application switch to the real servers must not contain a router. The router would stop HTTP requests from reaching the cache servers and, instead, direct them back out to the Internet.

More complex network topologies can be used if configuring IP proxy addresses (see [“IP Proxy Addresses for NAT” on page 252](#)).

For this example, the three cache real servers have the following IP addresses on the same IP subnet:

Table 11-1 Cache Redirection Example: Real Server IP Addresses

Cache Server	IP address
Server A	200.200.200.2
Server B	200.200.200.3
Server C	200.200.200.4

2. Install transparent cache software on all three cache servers.
3. Define an IP interface on the application switch.

The application switch must have an IP interface on the same subnet as the three cache servers because, by default, the application switch only remaps destination MAC addresses.

To configure an IP interface for this example, enter this command from the CLI:

```
>> # /cfg/ip/if 1                               (Select IP interface 1)
>> IP Interface 1# addr 200.200.200.100          (Assign IP address for the interface)
>> IP Interface 1# ena                           (Enable IP interface 1)
```

NOTE – The IP interface and the real servers must be in the same subnet. This example assumes that all ports and IP interfaces use default VLAN 1, requiring no special VLAN configuration for the ports or IP interface.

4. Define each real server on the switch.

For each cache real server, you must assign a real server number, specify its actual IP address, and enable the real server. For example:

```
>> ip# /cfg/slb/real 1                           (Server A is real server 1)
>> Real server 1# rip 200.200.200.2              (Assign Server A IP address)
>> Real server 1# ena                            (Enable real server 1)
>> Real server 1# ../real 2                      (Server B is real server 2)
>> Real server 2# rip 200.200.200.3              (Assign Server B IP address)
>> Real server 2# ena                            (Enable real server 2)
>> Real server 2# ../real 3                      (Server C is real server 3)
>> Real server 3# rip 200.200.200.4              (Assign Server C IP address)
>> Real server 3# ena                            (Enable real server 3)
```

5. Define a real server group.

This places the three cache real servers into one service group:

```
>> Real server 3# ../group 1                     (Select real server group 1)
>> Real server group 1# add 1                    (Add real server 1 to group 1)
>> Real server group 1# add 2                    (Add real server 2 to group 1)
>> Real server group 1# add 3                    (Add real server 3 to group 1)
```

6. Set the real server group metric to minmisses.

This setting helps minimize cache misses in the event real servers fail or are taken out of service:

```
>> Real server group 1# metric minmisses (Metric for minimum cache misses.)
```

7. Verify that server processing is disabled on the ports supporting application redirection.

NOTE – Do not use the “server” setting on a port with Application Redirection enabled. Server processing is used only with server load balancing. To disable server processing on the port, use the commands on the `/cfg/slb/port` menu, as described in the *Alteon OS Command Reference*.

8. Create a filter that will intercept and redirect all client HTTP requests.

The filter must be able to intercept all TCP traffic for the HTTP destination port and must redirect it to the proper port on the real server group:

```
>> SLB port 6# /cfg/slb/filt 2 (Select the menu for Filter 2)
>> Filter 2# sip any (From any source IP addresses)
>> Filter 2# dip any (To any destination IP addresses)
>> Filter 2# proto tcp (For TCP protocol traffic)
>> Filter 2# sport any (From any source port)
>> Filter 2# dport http (To an HTTP destination port)
>> Filter 2# action redir (Set the action for redirection)
>> Filter 2# rport http (Set the redirection port)
>> Filter 2# group 1 (Select real server group 1)
>> Filter 2# ena (Enable the filter)
```

The `rport` (redirection) parameter must be configured whenever TCP/UDP protocol traffic is redirected. The `rport` parameter defines the real server TCP or UDP port to which redirected traffic will be sent. The port defined by the `rport` parameter is used when performing Layer 4 health checks of TCP services.

Also, if NAT and proxy addresses are used on the application switch (see [Step 3 on page 245](#)), the `rport` parameter must be configured for all application redirection filters. Take care to use the proper port designation with `rport`: if the transparent proxy operation resides on the host, the well-known port 80 (or HTTP) is probably required. If the transparent proxy occurs on the application switch, make sure to use the service port required by the specific software package.

See [“IP Proxy Addresses for NAT” on page 252](#) for more information on IP proxy addresses.

9. Create a default filter.

In this case, the default filter will allow all noncached traffic to proceed normally:

>> Filter 2# ./filt 2048	<i>(Select the default filter)</i>
>> Filter 2048# sip any	<i>(From any source IP addresses)</i>
>> Filter 2048# dip any	<i>(To any destination IP addresses)</i>
>> Filter 2048# proto any	<i>(For any protocols)</i>
>> Filter 2048# action allow	<i>(Set the action to allow traffic)</i>
>> Filter 2048# ena	<i>(Enable the default filter)</i>

NOTE – When the `proto` parameter is not TCP or UDP, then `sport` and `dport` are ignored.

10. Assign the filters to the client ports.

Assuming that the redirected clients are connected to physical switch ports 5 and 6, both ports are configured to use the previously created filters as follows:

>> Filter 2048# ./port 5	<i>(Select the client port 5)</i>
>> SLB Port 5# add 2	<i>(Add filter 2 to port 5)</i>
>> SLB Port 5# add 2048	<i>(Add the default filter to port 5)</i>
>> SLB Port 5# filt enable	<i>(Enable filtering for port 5)</i>
>> SLB Port 5# ./port 6	<i>(Select the client port 6)</i>
>> SLB Port 6# add 2	<i>(Add filter 2 to port 6)</i>
>> SLB Port 6# add 2048	<i>(Add the default filter to port 6)</i>
>> SLB Port 6# filt enable	<i>(Enable filtering for port 6)</i>

11. Activate layer 4 services. Apply, and verify the configuration.

>> SLB Port 6# /cfg/slb	<i>(Select Server Load Balancing Menu)</i>
>> Layer 4# on	<i>(Activate Layer 4 software services)</i>
>> Layer 4# apply	<i>(Make your changes active)</i>
>> Layer 4# cur	<i>(View current settings)</i>

NOTE – SLB must be turned on in order for application redirection to work properly. The `on` command is valid only if the optional Layer 4 software is enabled on your application switch (see “Activating Optional Software” in the *Alteon OS Command Reference*).

12. Examine the resulting information from the `cur` command. If any settings are incorrect, make appropriate changes.

13. Save your new configuration changes.

```
>> Layer 4# save
```

(Save for restore after reboot)

14. Check the SLB information.

```
>> Layer 4# /info/slb
```

(View SLB information)

Check that all SLB parameters are working according to expectation. If necessary, make any appropriate configuration changes and then check the information again.

NOTE – Changes to filters on a given port only effect new sessions. To make filter changes take effect immediately, clear the session binding table for the port (see the `/oper/slb/clear` command in the *Alteon OS Command Reference*).

Delayed Binding for Cache Redirection

To configure delayed binding on your application switch for cache redirection only, use the following command:

```
>> # /cfg/slb/filt <filter number>/adv/urlp ena
```

For more conceptual information on delayed binding, see [“Delayed Binding” on page 167](#).

RTSP Cache Redirection

Alteon OS supports cache redirection for Real Time Streaming Protocol (RTSP). RTSP cache redirection is similar to HTTP cache redirection in configuration and in concept. Multimedia presentations consume a lot of Internet bandwidth. The quality of these presentations depends upon the real time delivery of the data. To ensure the high quality of multimedia presentations, several caching servers are needed to cache the multimedia data locally. This data is then made available quickly from the cache memory as required.

RTSP cache redirection redirects cached data transparently and balances the load among the cache servers. If there is no cache server, the request is directed to the origin server. Internet Service Providers use this feature to cache the multimedia data of a customer site locally. Since the requests for this data are directed to the local cache, they are served faster.

This section explains Layer 4 support for RTSP Streaming Cache Redirection. For detailed information on two prominent commercial RTSP servers—Real Player and QuickTime—see [“Real Time Streaming Protocol SLB” on page 176](#).

You can also configure the application switch to redirect client request based on URL content. For information on layer 7 RTSP Streaming Cache Redirection, see [“RTSP Streaming Cache Redirection” on page 456](#).

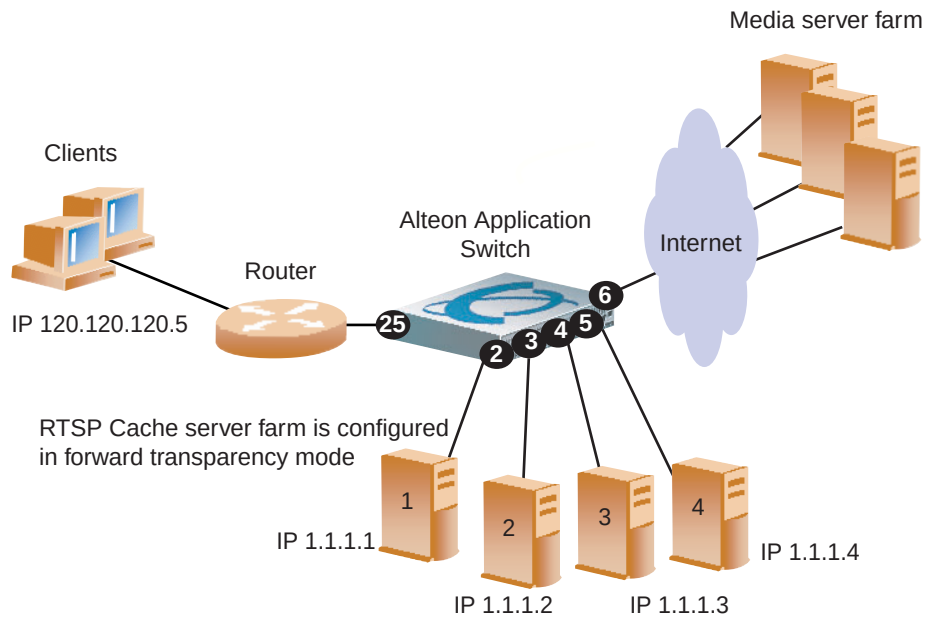


Figure 11-3 RTSP Cache Redirection

Follow this procedure to configure to load balance RTSP cache servers for the topology illustrated in [Figure 11-3](#):

1. Before configuring RTSP, do the following:

- Connect each cache server to the application switch
- Configure the IP addresses on all devices connected to the switch
- Configure the IP interfaces on the switch

2. At the application switch, configure RTSP cache servers and the IP addresses.

```
>> # /cfg/slb/real 1
>> Real server 1# rip 1.1.1.1           (Configure RTSP cache server 1)
>> Real server 1# ena                   (Enable RTSP cache server 1)
>> Real server 1# ../real 2
>> Real server 2# rip 1.1.1.2           (Configure RTSP cache server 2)
>> Real server 2# ena                   (Enable RTSP cache server 2)
>> Real server 2# ../real 3
>> Real server 3# rip 1.1.1.3           (Configure RTSP cache server 3)
>> Real server 3# ena                   (Enable RTSP cache server 3)
>> Real server 3# ../real 4
>> Real server 4# rip 1.1.1.4           (Configure RTSP cache server 4)
>> Real server 4# ena                   (Enable RTSP cache server 4)
```

3. Define a group to load balance the RTSP cache servers.

```
>> # /cfg/slb/group 1
>> Real Server Group 1# add 1           (Add RTSP cache server 1 to group 1)
>> Real Server Group 1# add 2           (Add RTSP cache server 2 to group 1)
>> Real Server Group 1# add 3           (Add RTSP cache server 3 to group 1)
>> Real Server Group 1# add 4           (Add RTSP cache server 4 to group 1)
```

4. Define the group metric for the RTSP cache servers.

RTSP supports all the standard load balancing metrics.

```
>>Real Server Group 1# metric leastconn (Set the metric to leastconn)
```

5. Configure an RTSP redirection filter to cache data and balance the load among the cache servers.

```
>> # /cfg/slb/filt 1                (Select the menu for filter 1)
>> Filter 1# action redir           (Set the action for redirection)
>> Filter 1# proto tcp              (Enter TCP protocol)
>> Filter 1# dport rtsp             (Enter service port for RTSP)
>> Filter 1# rport rtsp             (Enter redirection port for RTSP)
>> Filter 1# group 1                (Select RTSP cache server group 1)
>> Filter 1# adv                    (Select advanced menu for filter 1)
>> Filter 1# Advanced# proxy disable (Disable proxy)
```

6. Configure a default allow filter to facilitate traffic.

```
>> # /cfg/slb/filt 2048             (Select a default allow filter 2048)
>> Filter 2048# sip any             (From any source IP addresses)
>> Filter 2048# dip any             (To any destination IP addresses)
>> Filter 2048# ena                 (Enable a default allow filter)
>> Filter 2048# action allow        (Set the action to allow normal traffic)
```

7. Add and enable the redirection filter on the port to support basic cache redirection.

```
>> # /cfg/slb/port 25               (Select the menu for port 25)
>> SLB Port 25# add 1               (Add RTSP filter 1 to port 25)
>> SLB Port 25# add 2048           (Add default filter 2048 to port 25)
>> SLB Port 25# filt ena           (Enable filtering on port 25)
```

8. Apply and save the configuration.

```
>> SLB Port 25# apply
>> SLB Port 25# save
```

IP Proxy Addresses for NAT

Transparent proxies provide the benefits listed below when used with application redirection. Application redirection is automatically enabled when a filter with the `redir` action is applied on a port.

- With proxy IP addresses configured on ports that use redirection filters, the application switch can redirect client requests to servers located on any subnet.
- The application switch can perform transparent substitution for all source and destination addresses, including destination port remapping. This provides support for comprehensive, fully-transparent proxies. No additional client configuration is needed.

The following procedure can be used for configuring proxy IP addresses:

1. Configure proxy IP addresses and enable proxy for the redirection ports.

Each of the ports using redirection filters require proxy IP addresses. If Virtual Matrix Architecture (VMA) is enabled, you must configure all four proxy IP addresses. If VMA is disabled, only the client port with filters require proxy IP addresses. For more information on VMA and proxy IP addresses, see [“Virtual Matrix Architecture” on page 156](#) and [“Proxy IP Addresses” on page 156](#).

In this example, VMA is enabled, so configure four proxy IP addresses:

```
>> SLB port 3# /cfg/slb/pip                (Select proxy IP address menu)
>> Proxy IP Address# pip1 200.200.200.68   (Set proxy IP address)
>> Proxy IP Address# pip2 200.200.200.69   (Set proxy IP address)
>> Proxy IP Address# pip3 200.200.200.70   (Set proxy IP address)
>> Proxy IP Address# pip4 200.200.200.71   (Set proxy IP address)
>> Proxy IP Address# ../port 1              (Select port 1)
>> SLB port 1# proxy ena                    (Enable proxy port 1)
>> SLB port 1# ../port 2                    (Select port 2)
>> SLB port 2# proxy ena                    (Enable proxy port 2)
>> SLB port 2# ../port 3                    (Select port 3)
>> SLB port 3# proxy ena                    (Enable proxy port 3)
>> SLB port 3# ../port 4                    (Select port 4)
>> SLB port 4# proxy ena                    (Enable proxy port 4)
>> SLB port 4# ../port 5                    (Select port 5)
>> SLB port 5# proxy ena                    (Enable proxy port 5)
>> SLB port 5# ../port 6                    (Select port 6)
>> SLB port 6# proxy ena                    (Enable proxy port 6)
```

2. Configure the application redirection filters.

Once proxy IP addresses are established, configure each application redirection filter (Filter 2 in our example) with the real server TCP or UDP port to which redirected traffic will be sent. In this case, the requests are mapped to a different destination port (8080). You must also enable proxies on the real servers:

```
>> # /cfg/slb/filt 2                (Select the menu for Filter 2)
>> Filter 2# rport 8080             (Set proxy redirection port)
>> Filter 2# real 1/proxy enable     (Enable proxy on real servers)
>> Real server 1# ../real 2/proxy enable (Enable proxy on real servers)
>> Real server 2# ../real 3/proxy enable (Enable proxy on real servers)
```

NOTE – This configuration is not limited to HTTP (Web) service. Other TCP/IP services can be configured in a similar fashion. For example, if this had been a DNS redirect, `rport` would be sent to well-known port 53 (or the service port you want to remap to). For a list of other well-known services and ports, see the [Table 9-3 on page 148](#).

3. Apply and save your changes.

4. Check server statistics to verify that traffic has been redirected based on filtering criteria:

```
>> # /info/slb/group <group number>/filter <filter number>
```

Excluding Noncacheable Sites

Some sites provide content that is not well suited for redirection to cache servers. Such sites might provide browser-based games or applications that keep real-time session information or authenticate by client IP address.

To prevent such sites from being redirected to cache servers, create a filter that allows this specific traffic to pass normally through the application switch. This filter must have a higher precedence (a lower filter number) than the application redirection filter.

For example, if you want to prevent a popular Web-based game site on subnet 200.10.10.* from being redirected, you could add the following to the previous example configuration:

>> # /cfg/slb/filt 1	<i>(Select the menu for filter 1)</i>
>> Filter 1# dip 200.10.10.0	<i>(To the site's destination IP address)</i>
>> Filter 1# dmask 255.255.255.0	<i>(For entire subnet range)</i>
>> Filter 1# sip any	<i>(From any source IP address)</i>
>> Filter 1# proto tcp	<i>(For TCP traffic)</i>
>> Filter 1# dport http	<i>(To an HTTP destination port)</i>
>> Filter 1# sport any	<i>(From any source port)</i>
>> Filter 1# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 1# ena	<i>(Enable the filter)</i>
>> Filter 1# ../port 5	<i>(Select SLB port 5)</i>
>> SLB port 5# add 1	<i>(Add the filter to port 5)</i>
>> SLB port 5# ../port 6	<i>(Select SLB port 6)</i>
>> SLB port 6# add 1	<i>(Add the filter to port 6)</i>
>> SLB port 6# apply	<i>(Apply configuration changes)</i>
>> SLB port 6# save	<i>(Save configuration changes)</i>



CHAPTER 12

Health Checking

Health checking allows you to verify content accessibility in large Web sites. As content grows and information is distributed across different server farms, flexible, customizable content health checks are critical to ensure end-to-end availability.

The following Alteon OS health-checking topics are described in this chapter.

- **“Real Server Health Checks” on page 257.** This section explains the switch’s default health check, which checks the status of each service on each real server every two seconds.
- **“DSR Health Checks” on page 258.** This section describes the servers’ ability to respond to the client queries made to the Virtual server IP address when the server is in Direct Server Return (DSR) mode.
- **“Link Health Checks” on page 259.** This section describes how to perform Layer 1 health checking on an Intrusion Detection Server (IDS).
- **“TCP Health Checks” on page 260.** TCP health checks help verify the TCP applications that cannot be scripted.
- **“ICMP Health Checks” on page 260.** This section explains how ICMP health checks are used for UDP services.
- **“Script-Based Health Checks” on page 261.** This section describes how to configure the switch to send a series of health-check requests to real servers or real server groups and monitor the responses.
- **Application-based health checks:**
 - **“HTTP Health Checks” on page 267.** This section provides examples of HTTP-based health checks using hostnames.
 - **“UDP-Based DNS Health Checks” on page 269.** This section explains the functionality of the DNS Health Checks using UDP packets.
 - **“SNMP Health Check” on page 270.** This section explains how to perform SNMP health checks to real servers running SNMP Agents.

- [“FTP Server Health Checks” on page 271](#). This section describes how the File Transfer Protocol (FTP) server is used to perform health checks and explains how to configure the switch to perform FTP health checks.
- [“POP3 Server Health Checks” on page 272](#). This section explains how to use Post Office Protocol Version 3 (POP3) mail server to perform health checks between a client system and a mail server and how to configure the switch for POP3 health checks.
- [“SMTP Server Health Checks” on page 273](#). This section explains how to use Simple Mail Transfer Protocol (SMTP) mail server to perform health checks between a client system and a mail server and how to configure the switch for SMTP health checks.
- [“IMAP Server Health Checks” on page 274](#). This section describes how the mail server Internet Message Access Protocol (IMAP) protocol is used to perform health checks between a client system and a mail server.
- [“NNTP Server Health Checks” on page 275](#). This section explains how to use Network News Transfer Protocol (NNTP) server to perform health checks between a client system and a mail server and how to configure the switch for NNTP health checks
- [“RADIUS Server Health Checks” on page 276](#). This section explains how the RADIUS protocol is used to authenticate dial-up users to Remote Access Servers (RASs).
- [“HTTPS/SSL Server Health Checks” on page 277](#). This section explains how the switch queries the health of the SSL servers by sending an SSL client “Hello” packet and then verifies the contents of the server’s “Hello” response.
- [“WAP Gateway Health Checks” on page 277](#). This section discusses how the application switch provides a connectionless WSP health check for WAP gateways.
- [“LDAP Health Checks” on page 280](#). This section describes how to configure the switch to perform Lightweight Directory Access Protocol (LDAP) health checks for the switch to determine whether or not the LDAP server is running.
- [“ARP Health Checks” on page 282](#). This section describes how to perform health checks on Intrusion Detection Servers (IDS) that do not have full TCP/IP stack support.
- [“Failure Types” on page 283](#). This section explains the *service failed* and *server failed* states.

Real Server Health Checks

Alteon Application Switches running Server Load Balancing (SLB) monitor the servers in the real server group and the load-balanced application(s) running on them. If a switch detects that a server or application has failed, it will not direct any new connection requests to that server. When a service fails, an Alteon Application Switch can remove the individual service from the load-balancing algorithm without affecting other services provided by that server.

By default, the application switch checks the status of each service on each real server every two seconds. Sometimes, the real server may be too busy processing connections to respond to health checks. If a service does not respond to four consecutive health checks, the switch, by default, declares the service unavailable. You can modify both the health check interval and the number of retries.

>> # / cfg/slb/real <real server number>	<i>(Select the real server)</i>
>> Real server# inter 4	<i>(Check real server every 4 seconds)</i>
>> Real server# retry 6	<i>(If 6 consecutive health checks fail, declare real server down)</i>

NOTE – Health checks are performed sequentially when used in conjunction with a virtual server configured with multiple services and groups. As a result, the actual health-check interval could vary significantly from the value set for it using the `inter` parameter.

DSR Health Checks

Direct Server Return (DSR) health checks are used to verify the existence of a server-provided service where the server replies directly back to the client without responding through the virtual server IP address. In this configuration, the server will be configured with a real server IP address and virtual server IP address. The virtual server IP address is configured to be the same address as the user's virtual server IP address. When DSR health checks are selected, the specified health check is sent originating from one of the switch's configured IP interfaces, and is destined to the virtual server IP address with the MAC address that was acquired from the real server IP address's Address Resolution Protocol (ARP) entry.

Alteon OS allows you to perform health checks for DSR configurations. For more information, see [“Using Direct Server Return” on page 163](#). The switch is able to verify that the server correctly responds to requests made to the virtual server IP address as required in DSR configurations. To perform this function, the real server IP address is replaced with the virtual server IP address in the health check packets that are forwarded to the real servers for health checking. With this feature enabled, the health check will fail if the real server is not properly configured with the virtual server IP address.

NOTE – `viph1th` is enabled by default. This has no effect on the health check unless the real server is configured with DSR.

Configuring the Switch for DSR Health Checks

1. **Select the health check menu for a real server group.**

```
>> # /cfg/slb/group 1
```

(Select the Real Server Group 1 menu)

2. **Enable DSR VIP health checks for a real server group.**

For more information on DSR, see [“Using Direct Server Return” on page 163](#).

```
>> Real server group 1# viph1th enable|disable
```

3. **Apply and save your configuration.**

```
>> DSR VIP Health Check# apply
```

Link Health Checks

Link health checks are performed at the Layer 1 (physical) level, and are used on servers that do not respond to any other type of health check. Intrusion Detection Servers (IDSs) fall into this category.

The server is considered to be *up* when the link (connection) is present, and *down* when the link is absent. Many IDSs have two physical interfaces. One is used to detect intrusions, and the other is used to generate logging. The first interface detects intrusions but it does not have TCP/IP stack. So it is not possible to perform any health check other than Layer 1 health checking on the IDS. As long as the physical link between the switch and the IDS is up, it indicates the IDS is alive.

To perform this health check, a link option has been added to the real server group health command. The real server number is used to determine which port the server is connected to. For example, real server 1 is assumed to be connected to port 1. The valid IDS real server numbers are from 1 to 26 when health check is in use.

Configuring the Switch for Link Health Checks

Configure the switch to verify if the IDS server is alive by performing the following tasks:

1. **Select the health check menu for real server group 1.**

```
>> # /cfg/slb/group 1
```

2. **Set the health check type to link for real server group 1.**

```
>> # Real server group 1# health
Current health check type: tcp
Enter health check type: link
```

3. **Apply and save your configuration.**

```
>> # Real server group 1# apply
>> # Real server group 1# save
```

TCP Health Checks

TCP health checks are useful in verifying user-specific TCP applications that cannot be scripted.

Session switches monitor the health of servers and applications by sending Layer 4 connection requests (TCP SYN packets) for each load-balanced TCP service to each server in the server group on a regular basis. The rate at which these connection requests are sent is a user-configurable parameter. These connection requests identify both failed servers and failed services on a healthy server. When a connection request succeeds, the session switch quickly closes the connection by sending a TCP FIN (finished) packet.

NOTE – TCP health check is a default health check after you have configured the switch for a particular service.

ICMP Health Checks

Configure the switch with ICMP health check to verify if the real server is alive. The Layer 3 echo - echo reply health check is used for UDP services or when ICMP health checks are configured.

1. **Select the health check menu for group 1.**

```
>> # /cfg/slb/group 1
```

2. **Set the health check type to ICMP for group 1.**

```
>> # Real server group 1# health icmp
```

3. **Apply and save your configuration.**

```
>> # Real server group 1# apply
```

Script-Based Health Checks

The “send/expect” script-based health checks dynamically verify application and content availability using scripts. These scripts execute a sequence of tests to verify application and content availability.

Configuring the Switch for Script-Based Health Checks

You can configure the switch to send a series of health check requests to real servers or real server groups and monitor the responses. ASCII-based scripts can be used to verify application and content availability.

NOTE – Only TCP services can be health checked, since UDP protocols are usually not ASCII based.

The benefits of using script-based health checks are listed below:

- Ability to send multiple commands
- Check for any return string
- Test availability of different applications
- Test availability of multiple domains or Web sites

Alteon OS supports the following capacity for a single switch:

- 1024 bytes per script
- 16 scripts per switch
- approximately 10 to 15 health check statements (HTTP `get` and `expect` strings)

A simple command line interface controls the addition and deletion of ASCII commands to each script. New commands are added and removed from the end of the script. Commands exist to open a connection to a specific TCP port, send an ASCII request to the server, expect an ASCII string, and close a connection. The string configured with an `expect` command is searched for in each response packet. If it is not seen anywhere in any response packet before the real server health check interval expires, the server does not pass the expect step and fails the health check. A script can contain any number of these commands, up to the allowable number of characters that a script supports.

NOTE – Health check scripts can only be set up via the command line interface, but once entered, can be assigned as the health-check method using SNMP or the Browser-Based Interface (BBI).

Script Format

The general format for health-check scripts is shown below:

```
open application_port (e.g., 80 for HTTP, 23 for Telnet, etc.)
send request1
expect response1
send request2
expect response2
send request3
expect response3
close
```

NOTE – If you are doing HTTP 1.1 pipelining, you need to individually open and close each response in the script.

- Each script should start with the command **open port** <protocol port number>. The next line can be either a **send** or **expect**.
- The first word is the method. This is usually **get**; however, HTTP supports several other commands, including **put** and **head**. The second word indicates the content desired, or request-URI, and the third word represents the version of the protocol used by the client. If you supplied HTTP/1.1 for the protocol version, you would also have to add in the following line: **Host: www.hostname.com**

Example: **GET /index.html HTTP/1.1** (press Enter key)

Host: www.hostname.com (press Enter key twice)

This is known as a host header. It is important to include because most Web sites now require it for proper processing. Host headers were optional in HTTP/1.0 but are required when you use HTTP/1.1+.

- In order to tell the application server you have finished entering header information, a blank line of input is needed after all headers. At this point, the URL will be processed and the results returned to you.

NOTE – If you make an error, enter **rem** to remove the last typed script line entered. If you need to remove more than one line, enter **rem** for each line that needs to be removed.

- The switch provides the “\” prompt, which is one enter key stroke. When using the **send** command, note what happens when you type the **send** command with the command string. When you type **send**, press enter and allow the switch to format the command string (that is, \ versus \\\).

Scripting Guidelines

- Use generic result codes that are standard and defined by the RFC, as applicable. This helps ensure that if the server software changes, the servers won't start failing unexpectedly.
- Search only for the smallest and most concise piece of information possible. Each script cannot exceed 1K in size, so use the space wisely.
- Avoid tasks that may take a long time to perform or the health check will fail. For example, avoid tasks that exceed the interval for load balancing.

Script Configuration Examples

Script Example 1: A Basic Health Check

Configure the switch to check a series of Web pages (HTML or dynamic CGI scripts) before it declares a real server is available to receive requests.

NOTE – If you are using the CLI to create a health check script, you must use quotes (“”) to indicate the beginning and end of each command string.

```
/cfg/slb/group x/health script1/content none
/cfg/slb/adv/script1

open 80
send "GET /index.html HTTP/1.1\r\nHOST:www.hostname.com\r\n\r\n"
expect "HTTP/1.1 200"
close
open 80
send "GET /script.cgi HTTP/1.1\r\nHOST:www.hostname.com\r\n\r\n"
expect "HTTP/1.1 200"
close
open 443
...
close
```

NOTE – When you are using the command line interface to enter the `send` string as an argument to the `send` command, you must type two “\”s before an “n” or “r.” If you are instead prompted for the line, that is, the text string is entered after hitting <return>, then only one “\” is needed before the “n” or “r.”

Script Example 2: GSLB URL Health Check

In earlier Alteon OS releases, each remote Global Server Load Balancing site's virtual server IP address was required to be a real server of the local switch. Each switch sends a health check request to the other switch's virtual servers that are configured on the local switch. The health check is successful if there is at least one real server on the remote switch that is up. If all real servers on the remote switch are down, the remote real server (a virtual server of a remote switch) will respond with an HTTP Redirect message to the health check.

Using the scriptable health check feature, you can set up health check statements to check all the substrings involved in all the real servers.

Site 1 with Virtual Server 1 and the following real servers:

- Real Server 1 and Real Server 2: "images"
- Real Server 3 and Real Server 4: "html"
- Real Server 5 and Real Server 6: "cgi" and "bin"
- Real Server 7 (which is Virtual Server 2): "any"

Site 2 with Virtual Server 2 and the following real servers:

- Real Server 1 and Real Server 2: "images"
- Real Server 3 and Real Server 4: "html"
- Real Server 5 and Real Server 6: "cgi" and "bin"
- Real Server 7 (which is Virtual Server 2): "any"

A sample script is shown below:

```
/cfg/slb/group x/health script2/content none
/cfg/slb/adv/script2

open 80
send "GET /images/default.asp HTTP/1.1\r\nHOST: 192.192.1.2\r\n\r\n"
expect "HTTP/1.1 200"
close

open 80
send "GET /install/default.html HTTP/1.1\r\nHOST: 192.192.1.2\r\n\r\n"
expect "HTTP/1.1 200"
close

open 80
send "GET /script.cgi HTTP/1.1\r\nHOST: www.myurl.com \r\n\r\n"
expect "HTTP/1.1 200"
close
```


Script-based health checking is intelligent in that it will only send the appropriate requests to the relevant servers. In the example above, the first GET statement will only be sent to Real Server 1 and Real Server 2. Going through the health-check statements serially will ensure that all content is available by at least one real server on the remote site.

Configure the remote real server IP address (the virtual server IP address of the remote site) to accept “any” URL requests. The purpose of the first GET is to check if Real Server 1 or Real Server 2 is up—that is, to check if the remote site has at least one server for “images” content. Either Real Server 1 or Real Server 2 will respond to the first GET health check.

If all the real server IP addresses are down, Real Server 7 (the virtual server IP address of the remote site) will respond with an HTTP Redirect (respond code 302) to the health check. Thus, the health check will fail as the expected response code is 200, ensuring that the HTTP Redirect messages will not cause a loop.

Verifying Script-Based Health Checks

If a script fails, the expect line in the script that is not succeeding is displayed under the `/info/slb/real <real server number>` command:

```
>> # /info/slb/real 1
1: 205.178.13.225, 00:00:00:00:00:00, vlan 1, port 0, health 4, FAILED
  real ports:
    script 2, DOWN, current
      send GET / HTTP/1.0\r\n\r\n
      expect HTTP/1.0 200
```

The server is not responding to the `get` with the expect string.

When the script succeeds in determining the health of a real server, the following information is displayed:

```
>> # /info/slb/real 1
1: 205.178.13.223, 00:00:5e:00:01:24, vlan 1, port 2, health 4, up
  real ports:
    script 2, up, current
```

Application-Specific Health Checks

Application-specific health checks include the following applications:

- “HTTP Health Checks” on page 267
- “UDP-Based DNS Health Checks” on page 269
- “FTP Server Health Checks” on page 271
- “POP3 Server Health Checks” on page 272
- “SMTP Server Health Checks” on page 273
- “IMAP Server Health Checks” on page 274
- “NNTP Server Health Checks” on page 275
- “RADIUS Server Health Checks” on page 276
- “HTTPS/SSL Server Health Checks” on page 277
- “WAP Gateway Health Checks” on page 277
 - “WSP Content Health Checks” on page 278
 - “WTLS Health Checks” on page 279
- “LDAP Health Checks” on page 280

HTTP Health Checks

HTTP-based health checks can include the hostname for `HOST:` headers. The `HOST:` header and health check URL are constructed from the following components:

Item	Option	Configured Under	Max. Length
Virtual server hostname	hname	<code>/cfg/slb/virt/service</code>	9 characters
Domain name	dname	<code>/cfg/slb/virt</code>	35 characters
Server group health check field	content	<code>/cfg/slb/group</code>	34 characters

If the `HOST:` header is required, an HTTP/1.1 GET will occur. Otherwise, an HTTP/1.0 GET will occur. HTTP health check is successful if you get a return code of 200.

Example 1:

```
hname    = everest
dname    = alteonwebsystems.com
content  = index.html
```

Health check is performed using:

```
GET /index.html HTTP/1.1
Host: everest.alteonwebsystems.com
```

NOTE – If the content is not specified, the health check will revert back to TCP on the port that is being load balanced.

Example 2:

```
hname    = (none)
dname    = raleighduram.cityguru.com
content  = /page/gen/?_template=alteon
```

Health check is performed using:

```
GET /page/gen/?_template=alteon HTTP/1.1
Host: raleighduram.cityguru.com
```

Example 3:

```
hname    = (none)
dname    = jansus
content  = index.html
```

Health check is performed using:

```
GET /index.html HTTP/1.1
Host: jansus
```

Example 4:

```
hname    = (none)
dname    = (none)
content  = index.html
```

Health check is performed using:

```
GET /index.html HTTP/1.0 (since no HTTP HOST: header is required)
```

Example 5:

```
hname    = (none)
dname    = (none)
content  = //everest/index.html
```

Health check is performed using:

```
GET /index.html HTTP/1.1
Host: everest
```

Configuring the Switch for HTTP Health Checks

Perform the following on the switch to configure the switch for HTTP health checks:

1. Select the real server group.

```
>> # /cfg/slb/group 1 (Select a real server group)
```

2. Set the health check type to FTP for the real server group.

```
>> # /cfg/slb/group 1/health http
```

3. Configure the health check content.

```
>> # /cfg/slb/group 1/content <filename>
```

4. Apply and save your configuration.

```
>> # Real server group 1# apply
>> # Real server group 1# save
```

UDP-Based DNS Health Checks

Alteon OS supports UDP-based health checks along with TCP health checks, and performs load-balancing based on TCP and UDP protocols.

DNS servers can be based on both TCP and UDP protocols. With UDP-based DNS health checks enabled, you can send TCP-based queries to one real server group and UDP-based queries to another real server group.

The health check may be performed by sending a UDP-based query (for example, for `www.nortelnetworks.com`), and watching for the server's reply. The domain name to be queried may be modified by specifying the `content` command if you need to change the domain name.

Configuring the Switch for UDP-based Health Checks

Configure the switch to verify if the DNS server is alive.

1. **Select the real server group.**

```
>> # /cfg/slb/group 1
```

2. **Set the health check type to UDP for the real server group.**

```
>> # Real server group 1# health udpdns
```

3. **Set the content to domain name.**

```
>> # Real server group 1# content <filename> | /<host><filename> | none
```

4. **Apply and save your configuration.**

```
>> # Real server group 1# apply
>> # Real server group 1# save
```

NOTE – If no host name is configured, the health check is performed by sending a UDP-based query from a dummy host and watching for the server's reply. The reply, even though negative (for example, "Server not found" since the query is from a dummy host), serves the purpose of a health check, nonetheless.

SNMP Health Check

Alteon OS supports SNMP health checks by sending an SNMP GET request to the real server running an SNMP-based agent and watching for the server's reply.

You can configure up to five SNMP health checks and assign a group to use any one of them. For each SNMP health check, you configure the Object Identifier (OID) to be queried. SNMP health check is done by polling a single variable within the MIB. If the value returned by the real server for the MIB variable does not match the expected value configured in the `rcvcnt` field in the SNMP health check menu, then the server is marked down; the server is marked back up when it returns the expected value. By enabling the `invert` field, it is possible to configure the SNMP health check such that the server is marked down if the switch receives the expected value.

To configure the switch for SNMP health checks:

1. **Select the SNMP health check menu and enter an index number.**

```
>> # /cfg/slb/adv/snmphc 1 (Specify an index number)
```

2. **Specify the object identifier (OID).**

```
>> SNMP Health Check 1# oid 1.3.6.1.2.1.1.1.0
```

3. **Set the community string on the switch which notifies the SNMP agent on the real server to accept the SNMP packet from the switch.**

```
>> SNMP Health Check 1# comm real1
```

4. **Configure an integer value for the switch to accept the health check. Any other value that is returned from the real server is considered as health check failed.**

```
>> SNMP Health Check 1# rcvcnt 1
```

5. **Add the SNMP health check to a real server group.**

```
>> # /cfg/slb/group 10 (Select the real server group)
>> Real Server Group 10# health snmp1
```

If an SNMP health check fails the real server is marked failed under `/info/slb/real x`.

FTP Server Health Checks

The Internet File Transfer Protocol (FTP) provides facilities for transferring files to and from remote computer systems. Usually the user transferring a file needs authority to login and access files on the remote system. This protocol is documented in RFC 1123.

In normal Internet operation, the FTP server listens on the well-known port number 21 for control connection requests. The client sends a control message which indicates the port number on which the client is prepared to accept an incoming data connection request.

When a transfer is being set up, it is always initiated by the client. However, either the client or the server may be the sender of data. Along with transferring user requested files, the data transfer mechanism is also used for transferring directory listings from server to client.

NOTE – To configure the switch for FTP health checks, the FTP server must accept anonymous user login.

Configuring the Switch for FTP Health Checks

Create any file name from an FTP server under FTP server directory, for example, .txt, .exe, .bin and so forth.

To configure the switch for FTP health checks:

1. **Select the real server group.**

```
>> # /cfg/slb/group 1 (Select a real server group)
```

2. **Set the health check type to FTP for the real server group.**

```
>> # /cfg/slb/group 1/health ftp
```

3. **Configure the health check content.**

```
>> # /cfg/slb/group 1/content <filename>
```

4. **Apply and save your configuration.**

```
>> # Real server group 1# apply
>> # Real server group 1# save
```

POP3 Server Health Checks

The Post Office Protocol - Version 3 (POP3) is intended to permit a workstation to dynamically access a maildrop on a server host. The POP3 protocol is used to allow a workstation to retrieve mail that the server is holding for it. This protocol is documented in RFC 1939.

When the user on a client host wants to enter a message into the transport system, it establishes an SMTP connection to its relay host and sends all mail to it.

Initially, the server host starts the POP3 service by listening on TCP port 110. When a client host wants to make use of the service, it establishes a TCP connection with the server host.

Configuring the Switch for POP3 Health Checks

To support health checking on the UNIX POP3 server, the network administrator must configure a *username:password* value in the switch, using the `content` option on the SLB real server group menu (`/cfg/slb/group`)

To configure the switch for POP3 health checks:

1. **Select the real server group.**

```
>> # /cfg/slb/group 1
```

(Select a real server group)

2. **Set the health check type to POP3 for the real server group..**

```
>> # /cfg/slb/group 1/health pop3
```

3. **Configure the health check content.**

```
>> # /cfg/slb/group 1/content <username>:<password>
```

4. **Apply and save your configuration.**

```
>> # Real server group 1# apply
>> # Real server group 1# save
```


SMTP Server Health Checks

Simple Mail Transfer Protocol is a protocol to transfer e-mail messages between servers reliably and efficiently. This protocol traditionally operates over TCP, port 25 and is documented in RFC 821. Most e-mail systems that send mail over the Internet use SMTP to send messages from one server to another; the messages can then be retrieved with an e-mail client using either POP or IMAP.

Configuring the Switch for SMTP Health Checks

To support SMTP health checking, the network administrator must configure a *username:password* value in the switch, using the *content* option on the SLB real server group menu (*/cfg/slb/group*).

To configure the switch for SMTP health checks:

1. **Select the health check menu for the real server group.**

```
>> # /cfg/slb/group 1 (Select a real server group)
```

2. **Set the health check type to SMTP for the real server group..**

```
>> # /cfg/slb/group 1/health smtp
```

3. **Configure the health check content.**

```
>> # /cfg/slb/group 1/content <username>
```

4. **Apply and save your configuration.**

```
>> # Real server group 1# apply
>> # Real server group 1# save
```

IMAP Server Health Checks

Internet Message Access Protocol (IMAP) is a mail server protocol used between a client system and a mail server that allows a user to retrieve and manipulate mail messages. IMAP is not used for mail transfers between mail servers. IMAP servers listen to TCP port 143.

Configuring the Switch for IMAP Health Check

To support IMAP health checking, the network administrator must configure a *username:password* value in the switch, using the `content` option on the SLB Real Server Group Menu (`/cfg/slb/group`).

To configure the switch for IMAP health checks:

1. **Select the health check menu for the real server group.**

```
>> # /cfg/slb/group 1 (Select a real server group)
```

2. **Set the health check type to IMAP for the real server group..**

```
>> # /cfg/slb/group 1/health imap
```

3. **Configure the health check content.**

```
>> # /cfg/slb/group 1/content <username>:<password>
```

The `content` option specifies the *username:password* value that the server tries to match in its user database. In addition to verifying the user name and password, the database may specify the client(s) or port(s) the user is allowed to access.

4. **Apply and save your configuration.**

```
>> # Real server group 1# apply
>> # Real server group 1# save
```

NNTP Server Health Checks

Net News Transfer Protocol (NNTP) is a TCP/IP protocol based upon text strings sent bidirectionally over 7 bit ASCII TCP channels, and listens to port 119. It is used to transfer articles between servers as well as to read and post articles. NNTP specifies a protocol for the distribution, inquiry, retrieval, and posting of news articles using a reliable stream-based transmission of news among the ARPA-Internet community. NNTP is designed so that news articles are stored in a central database allowing a subscriber to select only those items he wishes to read.

NNTP is documented in RFC977. Articles are transmitted in the form specified by RFC1036.

Configuring the Switch for NNTP Health Checks

To configure the switch for NNTP health checks:

1. **Select the real server group.**

```
>> # /cfg/slb/group 1 (Select a real server group)
```

2. **Set the health check type to NNTP for the real server group.**

```
>> # /cfg/slb/group 1/health nntp
```

3. **Configure the health check content.**

```
>> # /cfg/slb/group 1/content <nntp newsgroup name>
```

Create nntp directory from MS Windows Option Pack4.

4. **Apply and save your configuration.**

```
>> # Real server group 1# apply
>> # Real server group 1# save
```

RADIUS Server Health Checks

The Remote Authentication Dial-In User Service (RADIUS) protocol is used to authenticate dial-up users to Remote Access Servers (RASs) and the client application they will use during the dial-up connection.

- **RADIUS Content Health Check Enhancements**
 - Include the switch IP as the Network-attached storage (NAS) IP parameter in the RADIUS content health check
 - RADIUS health check using the configured real server port (`rport`)
 - Variable-length RADIUS secret password. Supports less than 16 octets and up to 32 octets
- The `secret` value is a field of up to 32 alphanumeric characters used by the switch to encrypt a password during the RSA Message Digest Algorithm (MD5) and by the RADIUS server to decrypt the password during verification.
- The `content` option specifies the `username:password` value that the server tries to match in its user database. In addition to verifying the user name and password, the database may specify the client(s) or port(s) the user is allowed to access.

NOTE – Network-attached storage (NAS) is hard disk storage that is set up with its own network address rather than being attached to the department computer that is serving applications to a network's workstation users. By removing storage access and its management from the department server, both application programming and files can be served faster because they are not competing for the same processor resources. The network-attached storage device is attached to a local area network (typically, an Ethernet network) and assigned an IP address. File requests are mapped by the main server to the NAS file server.

Configuring the Switch for RADIUS Server Content Health Checks

The Alteon Application Switch will provide the NAS IP parameter while performing RADIUS content health checks. The switch uses the IP address of the IP interface that is on the same subnet as the RADIUS server or the default gateway as the NAS IP.

The RADIUS health check is performed using the configured real server port (`rport`). To configure RADIUS health checks, use the `/cfg/slb/virt <#>/service` menu.

Configuring the Switch for RADIUS Secret and Password

RADIUS is stateless and uses UDP as its transport protocol. To support RADIUS health checking, the network administrator must configure two parameters on the switch:

- the `/cfg/slb/secret` value
- the `content` parameter with a `username:password` value.

```
>> # /cfg/slb/group <real server group number> (Select the real server group)
>> # health radius (Specify the type of health checking)
>> # content <username>:<password> (Specify the RADIUS username:password value)
>> # /cfg/slb/adv/secret <RADIUS-coded value> (Enter up to 32 alphanumeric characters used to encrypt and decrypt password)
```

HTTPS/SSL Server Health Checks

The `sslh` health check option on the Real Server Group Menu (`/cfg/slb/group <#>`) allows the switch to query the health of the SSL servers by sending an SSL client “Hello” packet and then verify the contents of the server’s “Hello” response. SSL health check is performed using the real server port configured, that is, the `rport`.

The SSL enhanced health check behavior is summarized below:

- The switch sends a SSL “Hello” packet to the SSL server.
- If it is up and running, the SSL server responds with the “Server Hello” message.
- The switch verifies fields in the response and marks the service “Up” if the fields are OK.

During the handshake, the user and server exchange security certificates, negotiate an encryption and compression method, and establish a session ID for each session.

WAP Gateway Health Checks

Wireless Application protocol (WAP) carries Internet traffic to mobile devices and allows Web services to be delivered to mobile phones and handsets. The translation from HTTP/HTML to WAP/WML (Wireless Markup Language) is implemented by servers known as WAP gateways on the land-based part of the network. WAP devices can communicate in two ways:

- Wireless Session Protocol (WSP) content health checks, the unencrypted mode of sending WML traffic (similar to HTTPS).
- Wireless Transport Layer Security (WTLS) health checks, an encrypted mode of sending WML traffic (similar to HTTP).

WSP Content Health Checks

Wireless Session Protocol content health checks can be configured in two modes: connectionless and connection-oriented. Connectionless WSP runs on UDP/IP protocol, port 9200. Therefore, Alteon Application Switches can be used to load balance the gateways in this mode of operation.

Alteon OS provides a content-based health check mechanism where customized WSP packets can be sent to the gateways, and the switch can verify the expected response, in a manner similar to scriptable health checks.

The content of the WSP/UDP packet that is sent to the gateway can be configured as a hexadecimal string, which is encapsulated in a UDP packet and shipped to the server. Hence, this byte string should include all applicable WSP headers.

The content that the switch expects to receive from the gateway is also specified in the form of hexadecimal byte string. The switch matches each byte of this string with the received content.

If there is a mismatch of even a single byte on the received content, the gateway fails the health check. The user can also configure an offset for the received WSP packet: a byte index to the WSP response content from where the byte match can be performed.

Configuring the Switch for WSP Content Health Checks

1. **Select the WAP Health Check Menu.**

```
>> # /cfg/slb/adv/waphc
```

2. **Use the `sndcnt` command to enter the content to be sent to the WSP gateway.**

```
>> WAP Health Check# sndcnt
Current Send content:
Enter new Send content: 01 42 15 68 74 74 70 3a 2f 77 77 77 2e 6e 6f
6b 61 6d 00 .
```

3. **Enter the content that the switch expects to receive from the WSP gateway.**

```
>> WAP Health Check# rcvcnt
Current Receive content:
Enter new Receive content: 01 04 60 0e 03 94
```

NOTE – A maximum of 255 bytes of input are allowed on the switch command line. You may remove spaces in between the numbers to save space on the command line. For example, type **010203040506** instead of **01 02 03 04 05 06**.

4. Enter the WSP port.

```
>> WAP Health Check# wspport 9200
```

5. Set the offset value.

```
>> WAP Health Check# offset 0
```

6. Because WAP gateways are UDP-based and operate on a UDP port, configure UDP service in the virtual server menu.

```
>> # /cfg/slb/virt 1
>> Virtual Server 1# service (Configure virtual service 1)
Enter virtual port: 9200 (On the default WSP port)
>> Virtual Server 1 9200 Service# group 1 (Set the real server group number)
>> Virtual Server 1 9200 Service# udp ena (Enable UDP load balancing)
>> Virtual Server 1 9200 Service# apply (Apply the configuration)
```

7. Enable WSP health checks for group 1.

```
>> # /cfg/slb/group 1 (Select the Real Server Group 1 menu)
>> Real server group 1# health wsp (Set the health check type)
```

8. Apply and save the configuration.

```
>> Real server group 1# apply
```

WTLS Health Checks

Wireless Transport Layer Security (WTLS) can be configured to use ports 9202 and 9203 in connectionless and connection oriented modes respectively.

The WTLS health check feature provides a WTLS Hello based health check for connection-oriented WTLS traffic on port 9203. The application switch sends a new WTLS Client Hello to the WAP gateway, and checks to see if it receives a valid WTLS Server Hello back from the WAP Gateway.

Configuring the Switch for WTLS Health Checks

1. Select the group with the WAP gateway.

```
>> Main# /cfg/slb/group 1 (Select the Real Server Group 1 menu)
```

2. Use the `sndcnt` command to enter the content to be sent to the WSP gateway.

```
>> Real server group 1# health wtls
```

3. Select a port number other than 9203, if you want to change the port number on which your gateway is listening to WTLS traffic.

```
>> Main# /cfg/slb/adv/wap
>> WAP Health Check Menu # wtlsprt 10203
```

4. Apply and save your configuration.

```
>> WSP Health Check# apply
```

LDAP Health Checks

Lightweight Directory Access Protocol (LDAP) health checks enable the switch to determine whether the LDAP server is alive or not. LDAP versions 2 and 3 are described in RFC 1777 and RFC 2251.

The LDAP health check process consists of three LDAP messages over one TCP connection:

- **Bind request:** The switch first creates a TCP connection to the LDAP server on port 339, which is the default port. After the connection is established, the switch initiates an LDAP protocol session by sending an anonymous bind request to the server.
- **Bind response:** On receiving the bind request, the server sends a bind response to the switch. If the result code indicates that the server is alive, the switch marks the server as up. Otherwise, the switch marks the server as down even if the switch did this because the server did not respond within the timeout window.
- **Unbind request:** If the server is alive, the switch sends a request to unbind the server. This request does not require a response. It is necessary to send an unbind request as the LDAP server may crash if too many protocol sessions are active.

If the server is up, the switch closes the TCP connection after sending the unbind request. If the server is down, the connection is torn down after the bind response, if one arrives. The connection will also be torn down if it crosses the timeout limit, irrespective of the server's condition.

Configuring the Switch for LDAP Health Checks

Configure the switch to verify if the LDAP server is alive.

1. **Select the health check menu for the real server group.**

```
>> # /cfg/slb/group 1
```

2. **Set the health check type to LDAP for the real server group.**

```
>> # Real server group 1# health ldap
```

3. **Apply and save your configuration.**

```
>> # Real server group 1# apply
>> # Real server group 1# save
```

Determining the Version of LDAP

1. **Select the Advanced Menu.**

```
>> # Real server group 1# /cfg/slb/adv
```

2. **Set the version of LDAP. The default version is 2.**

```
>> # # Real server group 1# ldap <2 | 3> (Select the desired LDAP version)
```

3. **Apply and save your configuration.**

```
>> LDAP Health Check# apply
>> LDAP Health Check# save
```

ARP Health Checks

Address Resolution Protocol (ARP) is the TCP/IP protocol that resides within the Internet layer. ARP resolves a physical address from an IP address. ARP queries machines on the local network for their physical addresses. ARP also maintains IP to physical address pairs in its cache memory. In any IP communication, the ARP cache is consulted to see if the IP address of the computer or the router is present in the ARP cache. Then the corresponding physical address is used to send a packet.

In the switch, this feature allows the user to health check the Intrusion Detection Server (IDS) by sending an ARP query. The health check consists of the following sequence of actions:

1. **Accessing the ARP table.**
2. **Looking for the session entry in the ARP table. If the entry exists in the table, that means the real server is up, otherwise the health check has failed.**
3. **If the entry is present, then check the timestamp to find out if the last used time is greater than the ARP health check interval. If it is, then delete the query, as this means that the health check has failed.**
4. **Send another ARP request and repeat the above process until the timestamp shows the last used time smaller than the ARP health check interval.**

Configuring the Switch for ARP Health Checks

1. **Select the SLB group from the health check menu.**

```
>> /cfg/slb/group 1
```

2. **Select the type of health check to use.**

```
>> Real server group 1# health arp
```

3. **Enable ARP health checks for group 1.**

```
>> Real server group 1# arp
```

4. **Apply and save your configuration.**

```
>> Real server group 1# apply
```

Failure Types

Service Failure

If a certain number of connection requests for a particular service fail, the Alteon Application Switch places the service into the *service failed* state. While in this state, no new connection requests are sent to the server for this service; however, if graceful real server failure is enabled (`/cfg/slb/adv/grace ena`), state information about existing sessions is maintained and traffic associated with existing sessions continues to be sent to the server. Connection requests to, and traffic associated with, other load-balanced services continue to be processed by the server.

Example: A real server is configured to support HTTP and FTP within two real server groups. If a session switch detects an HTTP service failure on the real server, it removes that real server group from the load-balancing algorithm for HTTP, but keeps the real server in the mix for FTP. Removing only the failed service from load balancing allows users access to all healthy servers supporting a given service.

When a service on a server is in the *service failed* state, the Alteon Application Switch sends Layer 4 connection requests for the failed service to the server. When the session switch has successfully established a connection to the failed service, the service is restored to the load-balancing algorithm.

Server Failure

If all load-balanced services supported on a server fail to respond to switch connection requests within the specified number of attempts, then the server is placed in the *server failed* state. While in this state, no new connection requests are sent to the server; however, if graceful real server failure is enabled (`/cfg/slb/adv/grace ena`), state information about existing sessions is maintained and traffic associated with existing sessions continues to be sent to the server.

NOTE – All load-balanced services on a server must fail before the switch places the server in the *server failed* state.

The server is brought back into service as soon as the first service is proven to be healthy. Additional services are brought online as they are subsequently proven to be healthy.



CHAPTER 13

High Availability

Alteon Application Switches support high-availability network topologies through an enhanced implementation of the Virtual Router Redundancy Protocol (VRRP).

The following topics are discussed in this chapter:

- [“VRRP Overview” on page 286](#). This section discusses VRRP operation and Alteon OS redundancy configurations.
- [“Failover Methods” on page 292](#). This section describes the three modes of high availability.
- [“Alteon OS Extensions to VRRP” on page 298](#). This section describes VRRP enhancements implemented in Alteon OS.
- [“High Availability Configurations” on page 302](#). This section discusses a few of the more useful and easily deployed redundant configurations.
 - [“Active-Standby Virtual Server Router Configuration” on page 302](#)
 - [“Active-Active VIR and VSR Configuration” on page 304](#)
 - [“Active-Active Server Load Balancing Configuration” on page 306](#)
 - [“VRRP-Based Hot-Standby Configuration” on page 314](#)
- [“Virtual Router Deployment Considerations” on page 316](#). This section describes issues to consider when deploying virtual routers.
- [“Stateful Failover of Layer 4 and Layer 7 Persistent Sessions” on page 322](#). This section describes how to enable stateful failover by mirroring Layer 7 and Layer 4 persistent transactional states on the peer switch.

VRRP Overview

In a high-availability network topology, no device can create a single point-of-failure for the network or force a single point-of-failure to any other part of the network. This means that your network will remain in service despite the failure of any single device. To achieve this usually requires redundancy for all vital network components.

VRRP enables redundant router configurations within a LAN, providing alternate router paths for a host to eliminate single points-of-failure within a network. Each participating VRRP-capable routing device is configured with the same virtual router IP address and ID number. One of the virtual routers is elected as the master, based on a number of priority criteria, and assumes control of the shared virtual router IP address. If the master fails, one of the backup virtual routers will take control of the virtual router IP address and actively process traffic addressed to it.

Because the router associated with a given alternate path supported by VRRP uses the same IP address and MAC address as the routers for other paths, the host's gateway information does not change, no matter what path is used. A VRRP-based redundancy schema reduces administrative overhead because hosts need not be configured with multiple default gateways.

VRRP Components

Each physical router running VRRP is known as a *VRRP router*.

Virtual Interface Router

Two or more VRRP routers can be configured to form a *virtual interface router (VIR)*. (RFC 2338 calls this entity a *virtual router*.) The term virtual interface router will be used to distinguish this type of entity from a *virtual server router (VSR)*, as described in [“Alteon OS Extensions to VRRP” on page 298](#). When the term *virtual router* is used herein, the concept applies to both virtual interface routers and virtual server routers. Each VRRP router may participate in one or more virtual interface routers.

A virtual interface router acts as a default or next hop gateway for hosts on a LAN. Each virtual interface router consists of a user-configured *virtual router identifier (VRID)* and an IP address.

Virtual Router MAC Address

The VRID is used to build the *virtual router MAC Address*. The five highest-order octets of the virtual router MAC Address are the standard MAC prefix (00-00-5E-00-01) defined in RFC 2338. The VRID is used to form the lowest-order octet.

Owners and Renters

Only one of the VRRP routers in a virtual interface router may be configured as the IP address owner. This router has the virtual interface router's IP address as its real interface address. This router responds to packets addressed to the virtual interface router's IP address for ICMP pings, TCP connections, and so on.

There is no requirement for any VRRP router to be the IP address owner. Most VRRP installations choose not to implement an IP address owner. For the purposes of this chapter, VRRP routers that are not the IP address owner are called *renters*.

Master and Backup Virtual Router

Within each virtual router, one VRRP router is selected to be the virtual router master. See [“Selecting the Master VRRP Router” on page 289](#) for an explanation of the selection process.

NOTE – If the IP address owner is available, it will always become the virtual router master.

The virtual router master forwards packets sent to the virtual interface router. It also responds to Address Resolution Protocol (ARP) requests sent to the virtual interface router's IP address. Finally, the virtual router master sends out periodic advertisements to let other VRRP routers know it is alive and its priority.

Within a virtual router, the VRRP routers not selected to be the master are known as virtual router backups. Should the virtual router master fail, one of the virtual router backups becomes the master and assumes its responsibilities.

The Alteon Application Switches in [Figure 13-1](#) have been configured as VRRP routers. Together, they form a *virtual interface router (VIR)*.

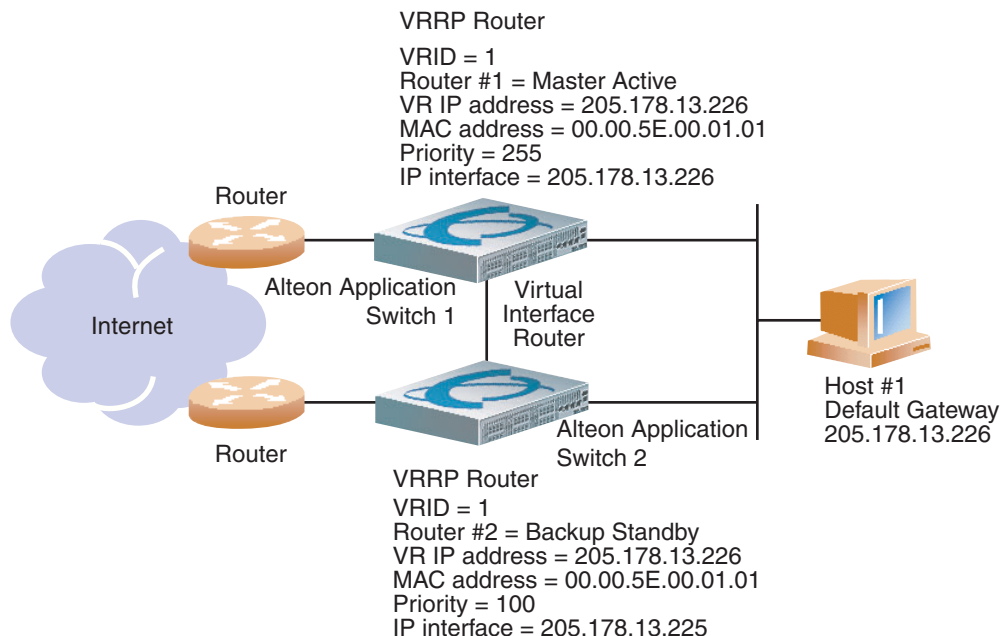


Figure 13-1 Example 1: VRRP Router

Application Switch 1 in [Figure 13-1](#) has its real interface configured with the IP address of the virtual interface router and is, therefore, the IP address owner. It also becomes the virtual router master. Application Switch 2 is a virtual router backup. Its real interface is configured with an IP address that is on the same subnet as the virtual interface router but is not the IP address of the virtual interface router.

The virtual interface router has been assigned a VRID = 1. Therefore, both of the VRRP routers have a MAC address = 00-00-5E-00-01-01.

VRRP Operation

The host shown in [Figure 13-1 on page 288](#) is configured with the virtual interface router's IP address as its default gateway. The master forwards packets destined to remote subnets and responds to ARP requests. In this example, the master is also the virtual interface router's IP address owner—therefore it also responds to ICMP ping requests and IP datagrams destined for the virtual interface router's IP address. The backup does not forward any traffic on behalf of the virtual interface router nor does it respond to ARP requests.

If the owner is not available, the backup becomes the master and takes over responsibility for packet forwarding and responding to ARP requests. However, because this switch is not the owner, it does not have a real interface configured with the virtual interface router's IP address.

Selecting the Master VRRP Router

Each VRRP router that is not an owner is configured with a priority between 1–254. According to the VRRP standard, an owner has a priority of 255. A bidding process determines which VRRP router is or becomes the master—the VRRP router with the highest priority. Owners have a higher priority than the range permitted for non-owners. If there is an IP address owner, it is always the master for the virtual interface router, as long as it is available.

The master periodically sends advertisements to an IP multicast address. As long as the backups receive these advertisements, they remain in the backup state. If a backup does not receive an advertisement for three advertisement intervals, it initiates a bidding process to determine which VRRP router has the highest priority and takes over as master.

If, at any time, a backup determines that it has higher priority than the current master does, it can preempt the master and become the master itself, unless configured not to do so. In preemption, the backup assumes the role of master and begins to send its own advertisements. The current master sees that the backup has higher priority and will stop functioning as the master.

A backup router can stop receiving advertisements for one of two reasons—the master can be down, or all communications links between the master and the backup can be down. If the master has failed, it is clearly desirable for the backup (or one of the backups, if there is more than one) to become the master.

NOTE – If the master is healthy but communication between the master and the backup has failed, there will then be two masters within the virtual router. To prevent this from happening, configure redundant links to be used between the switches that form a virtual router.

Active-Standby Failover

The previous text described the use of a group of VRRP routers to form a single virtual interface router. It implements a traditional *hot-standby* configuration in which the backup router only functions when the active router has failed. VRRP can also be used to implement *active-standby* configurations. In an active-standby configuration, both switches support *active* traffic but are configured so that they do not simultaneously support the same service.

In the example shown in [Figure 13-2](#), Application Switch 1 is the master for the virtual interface router with VRID = 1, and its backup for the virtual interface router with VRID = 2. Application Switch 2 is master for the virtual interface router with VRID = 2 and backup for the virtual interface router with VRID = 1. In this manner, both routers can actively forward traffic at the same time but not for the same interface.

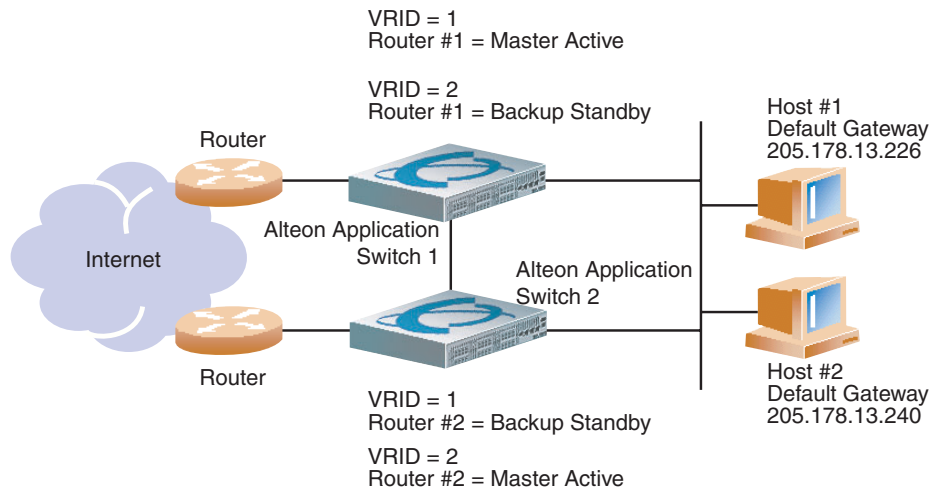


Figure 13-2 Example 2: VRRP Router

Table 13-1 Active Standby Configuration

	VRID = 1	VRID = 2
Application Switch 1	Router #1 = Master Active VR IP address = 205.178.13.226 MAC address = 00.00.5E.00.01.01 Priority = 255 IP interface = 205.178.13.226	Router #1 = Backup Standby VR IP address = 205.178.13.240 MAC address = 00.00.5E.00.01.02 Priority = 100 IP interface = 205.178.13.239
Application Switch 2	Router #2 = Backup Standby VR IP address = 205.178.13.226 MAC address = 00.00.5E.00.01.01 Priority = 100 IP interface = 205.178.13.225	Router #1 = Master Active VR IP address = 205.178.13.240 MAC address = 00.00.5E.00.01.02 Priority = 255 IP interface = 205.178.13.240

Failover Methods

With service availability becoming a major concern on the Internet, service providers are increasingly deploying Internet traffic control devices, such as application switches, in redundant configurations. Traditionally, these configurations have been *hot-standby* configurations, where one switch is active and the other is in a standby mode. A non-VRRP hot-standby configuration is shown in the figure below:

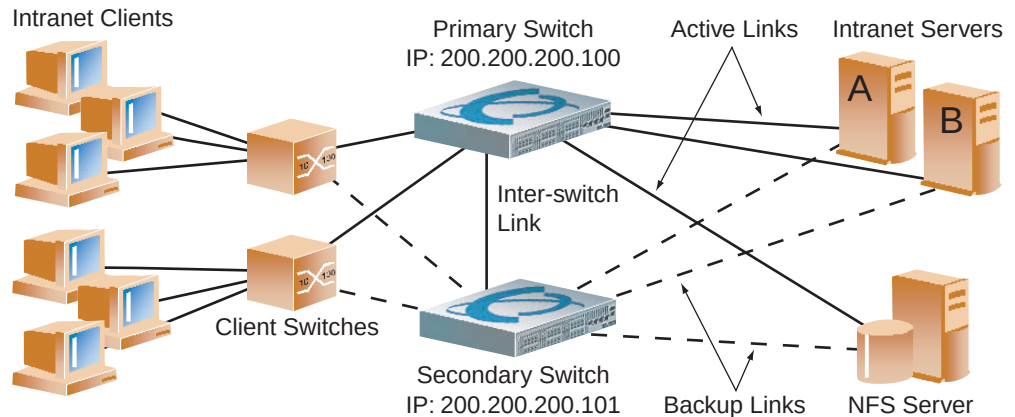


Figure 13-3 A Non-VRRP, Hot-Standby Configuration

While hot-standby configurations increase site availability by removing single points-of-failure, service providers increasingly view them as an inefficient use of network resources because one functional application switch sits by idly until a failure calls it into action. Service providers now demand that vendors' equipment support redundant configurations where all devices can process traffic when they are healthy, increasing site throughput and decreasing user response times when no device has failed.

Alteon OS high availability configurations are based on VRRP. The Alteon OS implementation of VRRP includes proprietary extensions to accommodate Layer 4 through Layer 7 application switching features.

The Alteon OS implementation of VRRP supports three modes of high availability:

- Active-Standby
- Active-Active
- Hot-Standby

The first mode, active-standby, is based on standard VRRP, as defined in RFC 2338. The second and third modes, *active-active* and *hot-standby*, are based on proprietary Alteon OS extensions to VRRP. Each mode is described in detail in the following sections.

Active-Standby Redundancy

In an active-standby configuration, shown in [Figure 13-4](#), two application switches are used. Both switches support active traffic but are configured so that they do not simultaneously support the same service. Each switch is active for its own set of services, such as IP routing interfaces or load-balancing virtual server IP addresses, and acts as a standby for other services on the other switch. If either switch fails, the remaining switch takes over processing for all services. The backup switch may forward Layer 2 and Layer 3 traffic, as appropriate.

NOTE – In an active-standby configuration, the same service cannot be active simultaneously on both switches.

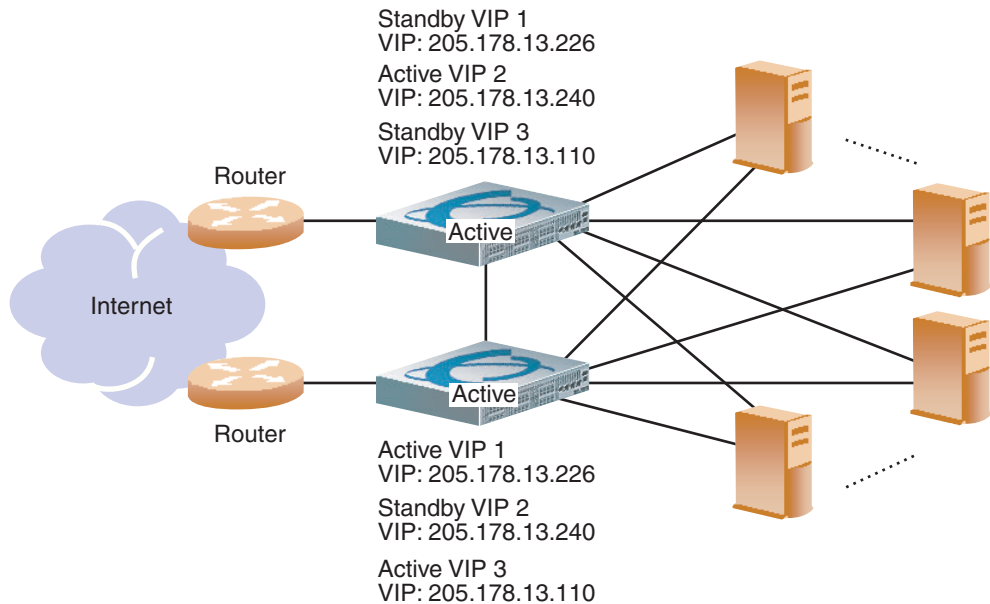


Figure 13-4 Active-Standby Redundancy

Active-Active Redundancy

In an active-active configuration, two application switches provide redundancy for each other, with both active at the same time for the same services.

Alteon OS has extended VRRP to include virtual servers, allowing full active/active redundancy between its Layer 4 switches. In an active-active configuration, shown in [Figure 13-5](#), both switches can process traffic for the same service at the same time; both switches can be active simultaneously for a given IP routing interface or load-balancing virtual server (VIP).

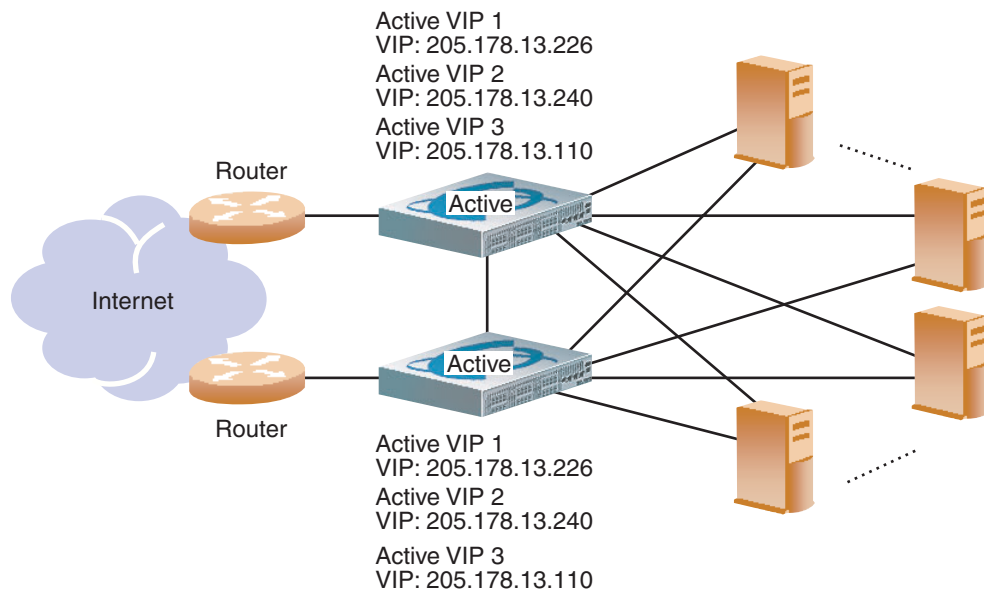


Figure 13-5 Active-Active Redundancy

In the example above, one switch is still the master router. However, traffic going through the backup router (associated with the same virtual router on the switch) that is addressed to the master router will be intercepted and processed by the backup router.

Hot-Standby Redundancy

In a hot-standby configuration, Spanning Tree Protocol (STP) is not needed to eliminate bridge loops. This speeds up failover when a switch fails. The standby switch blocks all ports configured as standby ports, whereas the master switch enables these same ports. Consequently, on a given switch, all virtual routers are either master or backup; they cannot change state individually.

To provide as much flexibility as possible, the old hot-standby approach has been modified to eliminate the problems previously associated with it and is now based on VRRP. In a hot-standby configuration, two or more switches provide redundancy for each other. One switch is elected *master* and actively processes Layer 4 traffic. The other switches (the backups) assume the master role should the master fail. The backups may forward Layer 2 and Layer 3 traffic as appropriate.

There are three components to the VRRP-based, hot-standby model: the virtual router group, additional Layer 4 port states, and configuration synchronization options. The hot-standby model is shown in [Figure 13-6](#).

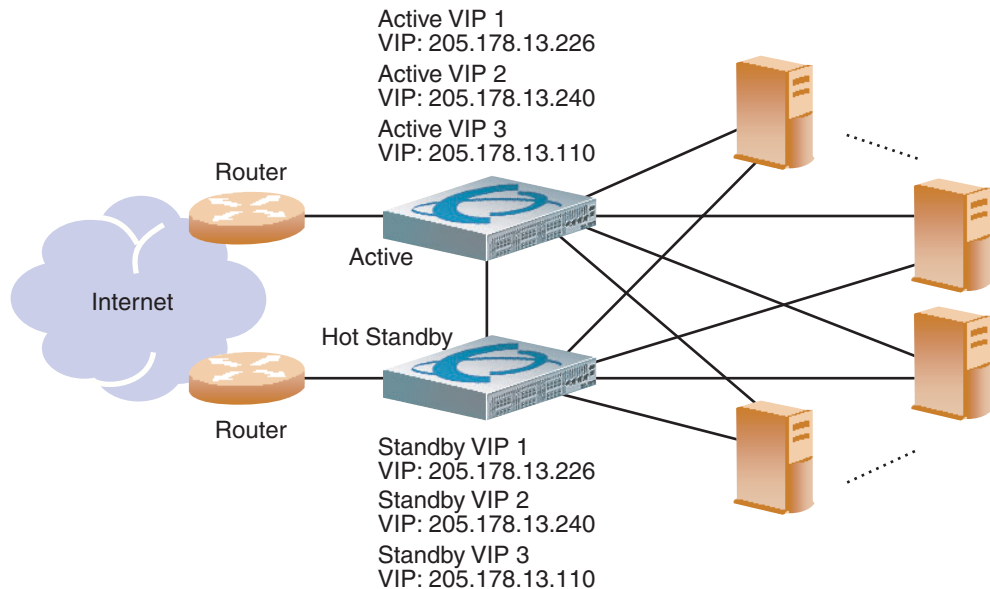


Figure 13-6 Hot-Standby Redundancy

Virtual Router Group

The virtual router group ties all of the virtual routers together as a single entity and is central to the hot-standby configuration. All virtual routers on a given switch must all be either master or backup. They cannot failover individually, only as a group. Once hot-standby is globally enabled, the virtual router group must be enabled. The virtual router group aggregates all of the virtual routers as a single entity.

If the virtual router group is master on one switch, it means the switch is master; otherwise, the switch is backup. However, Layer 4 processing is still enabled. If a virtual server is not a virtual router, the backup switch can still process traffic addressed to that virtual server IP address. Filtering is also still functional. Only traffic addressed to virtual server routers is not processed.

VRRP actually contains support for virtual router groups. Each advertisement is not limited to a single virtual router IP address and can include up to 256 addresses. This means that all virtual routers are advertised in the same packet, conserving processing and buffering resources.

Hot-Standby and Inter-Switch Port States

The second part of the solution involves introducing two additional Layer 4 port states, hot-standby and inter-switch:

- Links that attach to the standby switch must be configured as hot standby using `/cfg/slb/port x/hotstan`.
- Links that are used by VRRP to deliver updates are configured as `intersw`, or inter-switch links (not to be confused with Cisco's ISL). The command to configure one or more ports as interswitch links is `/cfg/slb/port <port number> /intersw`.

NOTE – A port cannot be configured to support both hot-standby and interswitch link.

The hot-standby switch listens to the master's VRRP updates. After an interval period has expired without receiving a update, the backup switch will take over. The forwarding states of hot-standby ports are controlled much like the forwarding states of the old hot-standby approach. Enabling hot-standby on a switch port allows the hot-standby algorithm to control the forwarding state of the port. If a switch is master, the forwarding states of the hot-standby ports are enabled. If a switch is backup, the hot-standby ports are blocked from forwarding or receiving traffic.

When the `hotstan` option (`/cfg/slb/port x/hotstan`) is enabled and all hot-standby ports have link, the virtual router group's priority is automatically incremented by the "track other virtual routers" value. This action allows the switches to failover when a hot-standby port

loses link. Other enabled tracking features only have affect when all hot-standby ports on a switch have link. The default virtual routers tracking value is two seconds. Keep in mind that this is an automatic process that cannot be turned off.

NOTE – The VRRP hot-standby approach does *not* support single-link failover. If one hot-standby port loses link, the entire switch must become master to eliminate loss of connectivity.

The forwarding states of non-hot-standby ports are not controlled via the hot-standby algorithm, allowing the additional ports on the switches to provide added port density. The client ports on both switches should be able to process or forward traffic to the master switch.

The inter-switch port state is only a place holder. Its presence forces the user to configure a inter-switch link when hot-standby is globally enabled and prohibits the inter-switch link from also being a hot-standby link for VRRP advertisements. These advertisements must be able to reach the backup switch.

Synchronizing Configurations

The final piece in configuring a high-availability solution includes the addition of synchronization options to simplify the manual configuration. Configuration options have been added to refine what is synchronized, to whom, and to disable synchronizing certain configurations. These options include proxy IP addresses, Layer 4 port configuration, filter configuration, and virtual router priorities.

Also, a peer menu (`cfg/slb/sync/peer`) has been added to allow the user to configure the IP addresses of the switches that should be synchronized. This provides added synchronization validation but does not require the users to enter the IP address of the redundant switch for each synchronization.

NOTE – When using both VRRP and GSLB, you must change the `/cfg/sys/wport` (Browser-Based Interface port) value of the target switch (the switch that is being synchronized to) to a port other than port 80 before VRRP synchronization begins.

For more information on synchronizing configurations between two switches, see [“Synchronizing Configurations” on page 321](#).

Alteon OS Extensions to VRRP

This section describes the following VRRP enhancements that are implemented in Alteon OS:

- [Virtual Server Routers](#)
- [Sharing and Active-Active Failover](#)
- [Tracking VRRP Router Priority](#)

Virtual Server Routers

Alteon OS supports *virtual server routers*, which extend the benefits of VRRP to virtual server IP addresses that are used to perform SLB.

Virtual server routers operate for virtual server IP addresses in much the same manner as Virtual Interface Routers operate for IP interfaces. A master is negotiated via a bidding process, during which information about each VRRP router's priority is exchanged. Only the master can process packets that are destined for the virtual server IP address and respond to ARP requests.

One difference between *virtual server routers* and *virtual interface routers* is that a virtual server router cannot be an *IP address owner*. All virtual server routers are *renters*.

All virtual routers, whether virtual server routers or virtual interface routers, operate independently of one another; that is, their priority assignments, advertisements, and master negotiations are separate. For example, when you configure a VRRP router's priority in a virtual server router, you are not affecting that VRRP router's priority in any virtual interface router or any other virtual server router of which it is a part. However, because of the requirement that MAC addresses be unique on a LAN, VRIDs must be unique among all virtual routers, whether virtual interface routers or virtual server routers.

Sharing and Active-Active Failover

Alteon OS supports *sharing* of interfaces at both Layer 3 and Layer 4, as shown in [Figure 13-7](#). With sharing, an IP interface or a VIP address can be active simultaneously on multiple switches, enabling active-active operation as shown in [Table 13-2](#).

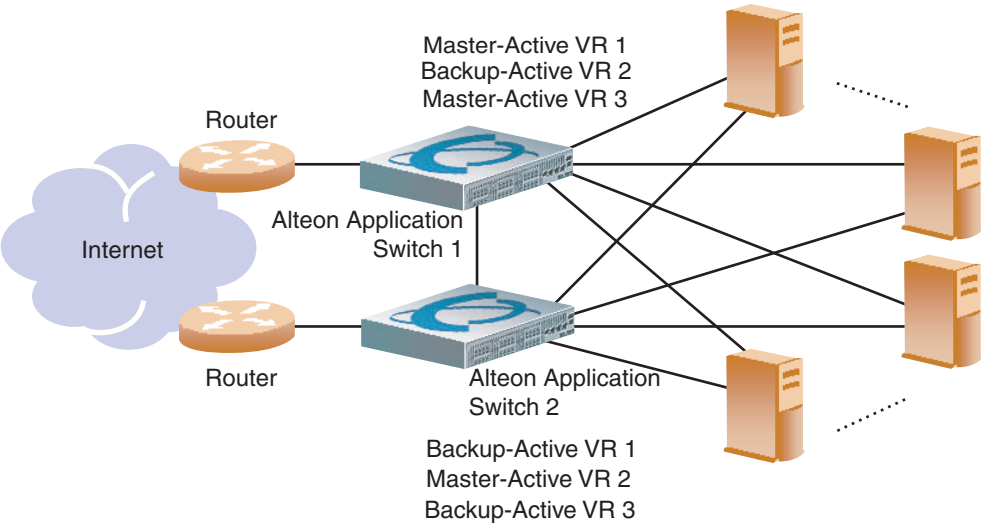


Figure 13-7 Active-Active High Availability

Table 13-2 Sharing Active-Active Failover

Application Switch 1	Master-Active VR 1 VRID 2 VIP: 205.178.13.226 MAC address 00-00-5E-OO-01-02	Backup-Active VR 2 VRID 4 VIP: 205.178.13.240 MAC address 00-00-5E-OO-01-04	Master-Active VR 3 VRID 6 VIP: 205.178.13.110 MAC address 00-00-5E-OO-01-06
Application Switch 2	Backup-Active VR 1 VRID 2 VIP: 205.178.13.226 MAC address 00-00-5E-OO-01-02	Master-Active VR 2 VRID 4 VIP: 205.178.13.240 MAC address 00-00-5E-OO-01-04	Backup-Active VR 3 VRID 6 VIP: 205.178.13.110 MAC address 00-00-5E-OO-01-06

When sharing is used, incoming packets are processed by the switch on which they enter the virtual router. The ingress switch is determined by external factors, such as routing and Spanning Tree configuration.

NOTE – Sharing cannot be used in configurations where incoming packets have more than one entry point into the virtual router—for example, where a hub is used to connect the switches.

When sharing is enabled, the master election process still occurs. Although the process does not affect which switch processes packets that must be routed or that are destined for the virtual server IP address, it does determine which switch sends advertisements and responds to ARP requests sent to the virtual router's IP address.

Alteon OS strongly recommends that sharing, rather than active-standby configurations, be used whenever possible. Sharing offers both better performance and fewer service interruptions in the face of fault conditions than active-standby configurations.

Tracking VRRP Router Priority

Alteon OS supports a tracking function that dynamically modifies the priority of a VRRP router, based on its current state. The objective of tracking is to have, whenever possible, the master bidding processes for various virtual routers in a LAN converge on the same switch. Tracking ensures that the selected switch is the one that offers optimal network performance. For tracking to have any effect on virtual router operation, preemption must be enabled.

NOTE – Tracking only affects hot standby and active-standby configurations. It does not have any effect on active-active sharing configurations.

Alteon OS can track the attributes listed in [Table 13-3](#):

Table 13-3 VRRP Tracking Parameters

Parameter	Description
Number of virtual routers in master mode on the switch <code>/cfg/vrrp/track/vrs</code>	Useful for ensuring that traffic for any particular client/server pair is handled by the same switch, increasing routing and load-balancing efficiency. This parameter influences the VRRP router's priority in both virtual interface routers and virtual server routers.
Number of IP interfaces active on the switch <code>/cfg/vrrp/track/ifs</code>	Helps elect the virtual routers with the most available routes as the master. (An IP interface is considered active when there is at least one active port on the same VLAN.) This parameter influences the VRRP router's priority in both virtual interface routers and virtual server routers.
Number of active ports on the same VLAN <code>/cfg/vrrp/track/ports</code>	Helps elect the virtual routers with the most available ports as the master. This parameter influences the VRRP router's priority in both virtual interface routers and virtual server routers.

Table 13-3 VRRP Tracking Parameters

Parameter	Description
Number of physical switch ports that have active Layer 4 processing on the switch <code>/cfg/vrrp/track/l4pts</code>	Helps elect the main Layer 4 switch as the master. This parameter influences the VRRP router's priority in both virtual interface routers and virtual server routers.
Number of healthy real servers behind the virtual server IP address that is the same as the IP address of the virtual server router on the switch <code>/cfg/vrrp/track/real</code>	Helps elect the switch with the largest server pool as the master, increasing Layer 4 efficiency. This parameter influences the VRRP router's priority in virtual server routers only.
In networks where the Hot Standby Router Protocol (HSRP) is used for establishing router failover, the number of Layer 4 client-only ports that receive HSRP advertisements <code>/cfg/vrrp/track/hsrp</code>	Helps elect the switch closest to the master HSRP router as the master, optimizing routing efficiency. This parameter influences the VRRP router's priority in both virtual interface routers and virtual server routers.

Each tracked parameter has a user-configurable weight associated with it. As the count associated with each tracked item increases (or decreases), so does the VRRP router's priority, subject to the weighting associated with each tracked item. If the priority level of a backup is greater than that of the current master, then the backup can assume the role of the master.

See [“Configuring the Switch for Tracking” on page 319](#) for an example on how to configure the switch for tracking VRRP priority.

High Availability Configurations

Alteon Application Switches offer flexibility in implementing redundant configurations. This section discusses a few of the more useful and easily deployed configurations:

- “Active-Standby Virtual Server Router Configuration” on page 302
- “Active-Active VIR and VSR Configuration” on page 304
- “Active-Active Server Load Balancing Configuration” on page 306
- “VRRP-Based Hot-Standby Configuration” on page 314

Active-Standby Virtual Server Router Configuration

Figure 13-8 shows an example configuration where two Alteon Application Switches are used as VRRP routers in an active-standby configuration, implementing a virtual server router. Active-standby redundancy should be used in configurations that cannot support sharing, that is, configurations where incoming packets will be seen by more than one switch, such as instances where a hub is used to connect the switches. In this configuration, when both switches are healthy, only the master responds to packets sent to the virtual server IP address.

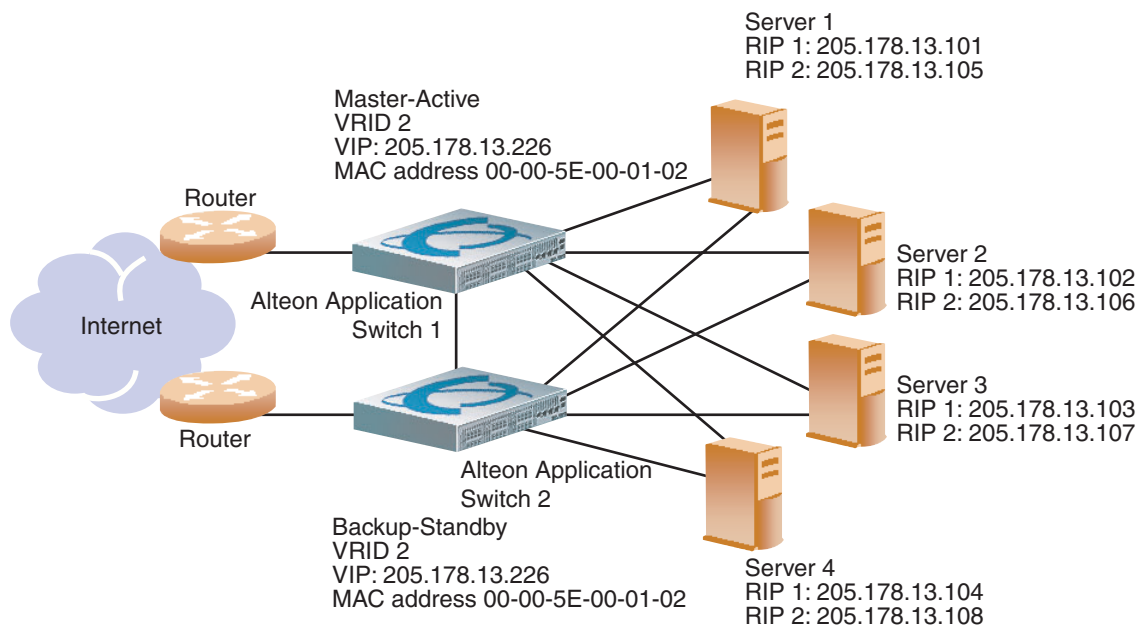


Figure 13-8 Active-Standby High-Availability Configuration

Although this example shows only two switches, there is no limit on the number of switches used in a redundant configuration. It is possible to implement an active-standby configuration across all the VRRP-capable switches in a LAN.

Each VRRP-capable switch in an active-standby configuration is autonomous. Switches in a virtual router need not be identically configured.

To implement the active-standby example, perform the following switch configuration:

- 1. Configure the appropriate Layer 2 and Layer 3 parameters on both switches.**

This includes any required VLANs, IP interfaces, default gateways, and so on. If IP interfaces are configured, none should use the virtual server IP address described in Step 4.

- 2. Define all filters required for your network configuration.**

Filters may be configured on one switch and synchronized with settings on the other switch (see [Step 5](#) below).

- 3. Configure all required SLB parameters on application switch 1.**

For the purposes of this example, assume that application switch 1 in [Figure 13-8](#) is configured in this step. Required Layer 4 parameters include a VIP = 205.178.13.226 and one real server group with four real servers, RIP = 205.178.13.101, RIP = 205.178.13.102, RIP = 205.178.13.103, and RIP = 205.178.13.104.

- 4. Configure the VRRP parameters on application switch 1.**

This configuration includes VRID = 2, VIP = 205.178.13.226 and the priority. Enable tracking and set the parameters appropriately (refer to [“Configuring the Switch for Tracking” on page 319](#)). Make sure to disable sharing.

- 5. Synchronize the SLB and VRRP configurations by synchronizing the configuration from application switch 1 to application switch 2.**

Use the `/oper/slb/sync` command (see [“Synchronizing Configurations” on page 321](#)).

- 6. Change the real servers in the application switch 2 configuration to RIP = 205.178.13.105, RIP = 205.178.13.106, RIP = 205.178.13.107, and RIP = 205.178.13.108.**

Adjust application switch 2's priority (see [“Configuring the Switch for Tracking” on page 319](#)).

In this example, with application switch 1 as the master, if a link between application switch 1 and a server fails, the server will fail health checks and be taken out of the load-balancing algorithm. If tracking is enabled and is configured to take into account the number of healthy real servers for the Virtual Router's VIP address, application switch 1's priority will be reduced. If it is reduced to a value lower than application switch 2's priority, then application switch 2 will assume the role of master. In this case, all active connections serviced by application switch 1's virtual server IP address are severed.

If the link between application switch 1 and its Internet router fails, the protocol used to distribute traffic between the routers, for example, Open Shortest Path First (OSPF), will reroute traffic to the other router. application switch 2 (backup) will act as a Layer 2/3 switch and forward all traffic destined to the virtual server IP address to application switch 1.

If the entire application switch 1 (master) fails, the protocol used to distribute traffic between the routers, such as OSPF, will reroute traffic to application switch 2. Application switch 2 (backup) detects that the master has failed because it will stop receiving advertisements. The backup then assumes the master's responsibility of responding to ARP requests and issuing advertisements.

Active-Active VIR and VSR Configuration

Figure 13-9 two Alteon Application Switches are used as VRRP routers in an active-active configuration implementing a virtual server router. As noted earlier, this is the preferred redundant configuration.

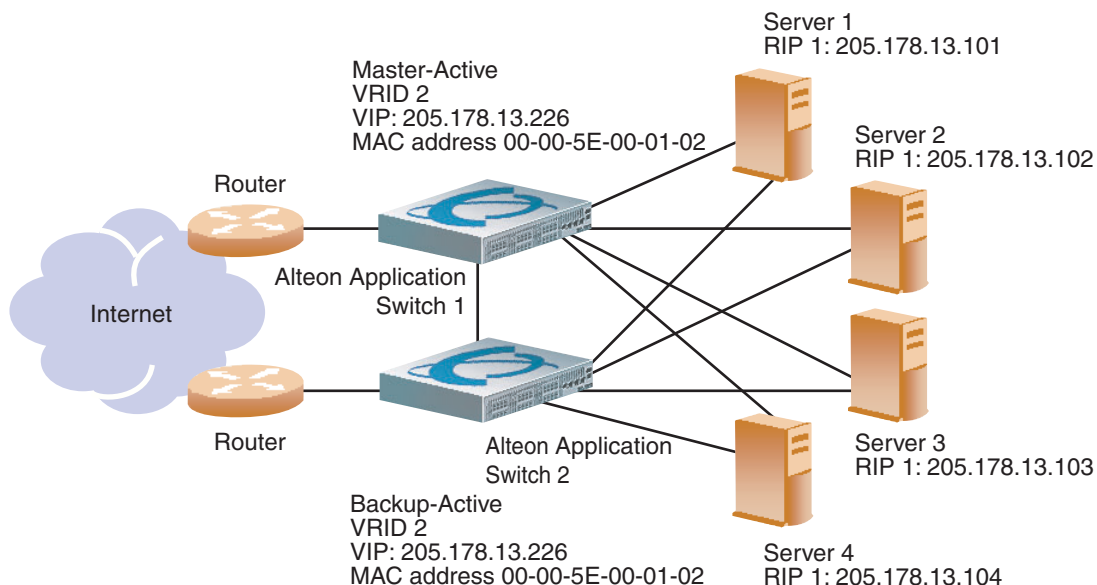


Figure 13-9 Active-Active High-Availability Configuration

Although this example shows only two switches, there is no limit on the number of switches used in a high availability configuration. It is possible to implement an active-active configuration and perform load sharing between all of the VRRP-capable switches in a LAN.

In this configuration, when both switches are healthy, the load balanced packets are sent to the virtual server IP address, resulting in higher capacity and performance than when the switches are used in an active-standby configuration.

The switch on which a frame enters the virtual server router is the one that processes that frame. The ingress switch is determined by external factors, such as routing and STP settings.

NOTE – Each VRRP-capable switch is autonomous. There is no requirement that the switches in a virtual router be identically configured. Different switch models with different numbers of ports and different enabled services may be used in a virtual router.

To implement this example, configure the switches as follows:

1. Configure the appropriate Layer 2 and Layer 3 parameters on both switches.

This configuration includes any required VLANs, IP interfaces, default gateways, and so on. If IP interfaces are configured, none of them should use the VIP address described in Step 4.

2. Define all filters required for your network configuration.

Filters may be configured on one switch and synchronized with the settings on the other switch (see [Step 5](#), below).

3. Configure all required SLB parameters on one of the switches.

For the purposes of this example, assume that application switch 1 (see [Figure 13-9 on page 304](#)) is configured in this step. Required Layer 4 parameters include a VIP = 205.178.13.226 and one real server group with two real servers, RIP = 205.178.13.101 and RIP = 205.178.13.102.

RIP = 205.178.13.103 should be configured as a backup server to RIP = 205.178.13.101.

RIP = 205.178.13.104 should be configured as a backup server to RIP = 205.178.13.102.

NOTE – In this configuration, each server's backup is attached to the other switch. This ensures that operation will continue if all of the servers attached to a switch fail.

4. Configure the VRRP parameters on the switch.

Configure VRID = 2, VIP address = 205.178.13.226, and priority. Be sure to enable sharing.

5. Synchronize the SLB and VRRP configurations by pushing the configuration from application switch 1 to application switch 2.

Use the `/oper/slb/sync` command.

6. Reverse the roles of the real servers and their backups in application switch 2's configuration.

Configure RIP = 205.178.13.103 and RIP= 205.178.13.104 as the real servers

Configure RIP = 205.178.13.101 and RIP = 205.178.13.102 as their backups, respectively.

In this configuration, if a link between a switch and a server fails, the server will fail health checks and its backup (attached to the other switch) will be brought online. If a link between a switch and its Internet router fails, the protocol used to distribute traffic between the routers (for example, OSPF) will reroute traffic to the other router. Since all traffic now enters the virtual server router on one switch, that switch will process all incoming connections.

If an entire master switch fails, the backup will detect this failure because it will stop receiving advertisements. The backup will assume the master's responsibility of responding to ARP requests and issuing advertisements.

Be cautious before setting the maximum connections (`maxconn`) metric in this configuration. The `maxcon` number is not shared between switches. Therefore, if a server is used for normal operation by one switch and is activated simultaneously as a backup by the other switch, the total number of possible connections to that server will be the sum of the maximum connection limits defined for it on both switches.

Active-Active Server Load Balancing Configuration

In this example, you set up four virtual servers each load balancing two servers providing one service (for example, HTTP) per virtual server.

You are load balancing HTTP, HTTPS, POP3, SMTP, and FTP. Each protocol is load balanced via a different virtual server. You could load balance all of these services on one VIP, but in this example, four distinct virtual servers are used to illustrate the benefits of active/active failover. Set up one switch, dump out the configuration script (also called a text dump), edit it, and dump the edited configuration into the peer switch.

NOTE – Configuring the switch for active-active failover should take no longer than 15 minutes to complete. You can use either the Alteon OS Browser-Based Interface (BBI) or the Command Line Interface (CLI) for configuration.

Task 1: Background Configuration

1. Define the IP interfaces.

The switch will need an IP interface for each subnet to which it will be connected so it can communicate with devices attached to it. Each interface will need to be placed in the appropriate VLAN. In our example, Interfaces 1, 2, 3, and 4 will be in VLAN 2 and Interface 5 will be in VLAN 3.

NOTE – On Alteon Application Switches, you may configure more than one subnet per VLAN.

To configure the IP interfaces for this example, enter the following commands from the CLI:

>> Main# / cfg/ip/if 1	(Select IP interface 1)
>> IP Interface 1 # addr 10.10.10.10	(Assign IP address for the interface)
>> IP Interface 1 # vlan 2	(Assign VLAN for the interface)
>> IP Interface 1 # ena	(Enable IP interface 1)

Repeat the commands for each interface listed below:

- IF 2—20.10.10.10
- IF 3—30.10.10.10
- IF 4—40.10.10.10
- IF 5—200.1.1.10

2. Define the VLANs.

In this configuration, set up two VLANs: One for the outside world (the ports connected to the upstream switches, toward the routers) and one for the inside (the ports connected to the downstream switches, toward the servers).

>> Main# / cfg/vlan <VLAN number>	(Select VLAN 3)
>> vlan 3 # add <port number>	(Add a port to the VLAN membership)
>> vlan 3 # ena	(Enable VLAN 3)

Repeat this command for the second VLAN.

- VLAN 3 - IF 5—physical ports connected to upstream switches.
- VLAN 2 - IFs 1,2,3,4—physical ports connected to downstream switches.

3. Disable Spanning Tree.

>> Main# / cfg/stp 1	(Select the STP Group number)
>> Main# / cfg/stp 1/ off	(Disable STP)
>> Main# / cfg/stp 1/ apply	(Make your changes active)

4. Enable IP forwarding.

IP forwarding is enabled by default. Make sure IP forwarding is enabled if the virtual server IP addresses and real server IP addresses are on different subnets, or if the switch is connected to different subnets and those subnets need to communicate through the switch. If you are in

doubt as to whether or not to enable IP forwarding, enable it. In this example, the virtual server IP addresses and real server IP addresses are on different subnets, so enable this feature using the following command:

```
>> Main# /cfg/ip/frwd/on (Enable IP forwarding)
```

Task 2: SLB Configuration

1. Define the Real Servers.

The real server IP addresses are defined and put into four groups, depending on the service they are running. Notice that RIPs 7 and 8 are on routable subnets in order to support passive FTP. For each real server, you must assign a real server number, specify its actual IP address, and enable the real server.

For example:

```
>> Main# /cfg/slb/real 1 (Server A is real server 1)
>> Real server 1 # rip 10.10.10.5 (Assign Server A IP address)
>> Real server 1 # ena (Enable real server 1)
```

Repeat this sequence of commands for the following real servers:

- RIP 2—10.10.10.6/24
- RIP 3—20.10.10.5/24
- RIP 4—20.10.10.6/24
- RIP 5—30.10.10.5/24
- RIP 6—30.10.10.6/24
- RIP 7—200.1.1.5/24
- RIP 8—200.1.1.6/24

2. Define the real server groups, adding the appropriate real servers.

This combines the three real servers into one service group:

```
>> Real server 8 # /cfg/slb/group 1 (Select real server group 1)
>> Real server group 1# add 1 (Add real server 1 to group 1)
>> Real server group 1# add 2 (Add real server 2 to group 1)
```

Repeat this sequence of commands for the following real server groups:

- Group 2—Add RIP 3 and 4
- Group 3—Add RIP 5 and 6
- Group 4—Add RIP 7 and 8

3. Define the virtual servers.

After defining the virtual server IP addresses and associating them with a real server group number, you must tell the switch which IP ports/services/sockets you want to load balance on each VIP. You can specify the service by either the port number, service name, or socket number.

```
>> Real server group 4 # /cfg/slb/virt 1    (Select virtual server 1)
>> Virtual server 1 # vip 200.200.200.100  (Assign a virtual server IP address)
>> Virtual Server 1 # service 80           (Assign HTTP service port 80)
>> Virtual server 1 http Service # group 1 (Associate virtual port to real group)
>> Virtual server 1 # ena                  (Enable the virtual server)
```

Repeat this sequence of commands for the following virtual servers:

- VIP 2—200.200.200.101 will load balance HTTPS (Port 443) to Group 2
- VIP 3—200.200.200.102 will load balance POP/SMTP (Ports 110/25) to Group 3
- VIP 4—200.200.200.104 will load balance FTP (Ports 20/21) to Group 4

4. Define the client and server port states.

Defining a client port state tells that port to watch for any frames destined for the VIP and to load balance them if they are destined for a load-balanced service. Defining a server port state tells the port to do the remapping (NAT) of the real server IP address back to the virtual server IP address. Note the following:

- The ports connected to the upstream switches (the ones connected to the routers) will need to be in the client port state.
- The ports connected to the downstream switches (the ones providing fan out for the servers) will need to be in the server port state.

Configure the ports, using the following sequence of commands:

```
>> Virtual server 4# /cfg/slb/port 1        (Select physical switch port 1)
>> SLB port A1 # client ena                (Enable client processing on port 1)
>> SLB port A1 # ../port 2                 (Select physical switch port 2)
>> SLB port A2 # server ena                (Enable server processing on port 2)
```

Task 3: Virtual Router Redundancy Configuration

1. Configure virtual routers 2, 4, 6, and 8.

These virtual routers will have the same IP addresses as the virtual server IP address. This is what tells the switch that these are virtual service routers (VSRs). In this example, Layer 3 bindings are left in their default configuration, which is disabled.

Configure a virtual router using the following sequence of commands:

```
>> Virtual server 4 # /cfg/vrrp/vr 2      (Select virtual router 2)
>> Virtual router 2 vrid 2              (Set virtual router ID)
>> Virtual router 2 addr 200.200.200.100 (Assign virtual router IP address)
>> Virtual router 2 if 5                 (Assign virtual router interface)
>> Virtual router 2 ena                  (Enable virtual router 2)
```

Repeat this sequence of commands for the following virtual routers:

- VR 4 - VRID 4 - IF 5 (associate with IP interface #5)—Address 200.200.200.101
- VR 6 - VRID 6 - IF 5 (associate with IP interface #5)—Address 200.200.200.103
- VR 8 - VRID 8 - IF 5 (associate with IP interface #5)—Address 200.200.200.104

2. Configure virtual routers 1, 3, 5, and 7.

These virtual routers will act as the default gateways for the servers on each respective subnet. Because these virtual routers are survivable next hop/default gateways, they are called virtual interface routers (VIRs).

Configure each virtual router listed below, using the sequence of commands in Step 1.

- VR 1 - VRID 1 - IF 1 (associate with IP interface 1)—Address 10.10.10.1
- VR 3 - VRID 3 - IF 2 (associate with IP interface 2)—Address 20.10.10.1
- VR 5 - VRID 5 - IF 3 (associate with IP interface 3)—Address 30.10.10.1
- VR 7 - VRID 7 - IF 4 (associate with IP interface 4)—Address 40.10.10.1

3. Set the renter priority for each virtual router.

Since you want Switch 1 to be the master router, you need to bump the default virtual router priorities (which are 100 to 101 on virtual routers 1-4) to force switch 1 to be the master for these virtual routers.

Use the following sequence of commands:

```
>> Virtual server 4 # /cfg/vrrp/vr 1      (Select virtual router 1)
>> Virtual router 1 prio 101             (Set virtual router priority)
```

Apply this sequence of commands to the following virtual routers, assigning each a priority of 101:

- VR 2—Priority 101
- VR 3—Priority 101
- VR 4—Priority 101

4. Configure priority tracking parameters for each virtual router.

For this example, the best parameter(s) on which to track is Layer 4 ports (l4pts).

Use the following command:

```
>> Virtual server 4# /cfg/vrrp/vr 1/track l4pts
```

This command sets the priority tracking parameter for virtual router 1, electing the virtual router with the most available ports as the master router. Repeat this command for the following virtual routers:

- VR 2 - Track l4pts VR 6 - Track l4pts
- VR 3 - Track l4pts VR 7 - Track l4pts
- VR 4 - Track l4pts VR 8 - Track l4pts

Switch 1 configuration is complete.

Task 4: Configuring Switch 2

Use the following procedure to dump the configuration script (text dump) out of Switch 1:

- Using the Browser Based Interface (BBI)
 - (a) You need a serial cable that is a DB-9 Male to DB-9 Female, straight-through (not a null modem) cable.
 - (b) Connect the cable from a COM port on your computer to the console port on switch 1.
 - (c) Open HyperTerminal (or the terminal program of your choice) and connect to the switch using the following parameters: Baud: 9600, Data Bits: 8, Parity: None, Stop Bits:1, Flow Control: None.
- Using HyperTerminal
 - (a) Only the Baud Rate and Flow Control options need to be changed from the default settings.
 - (b) Once you connect to the switch, start logging your session in HyperTerminal (transfer/capture text).
 - (c) Save the file as “Customer Name” Switch 1, then type the following command in the switch command line interface:


```
/cfg/dump
```

A script will be dumped out.
 - (d) Stop logging your session (transfer/capture text/stop).

Modify the script as follows:

1. Open the text file that you just created and change the following:

- Delete anything above “Script Start.”
- Delete the two lines directly below “Script Start.” These two lines identify the switch from which the dump was taken and the date and time. If these two lines are left in, it will confuse Application Switch 2 when you dump in the file.
- Change the last octet in all the IP interfaces from .10 to .11. Find this in line: `/cfg/ip/if 1/addr 10.10.10.10`. Simply delete the “0” and put in a “1.” Be sure to do this for all the IP interfaces, otherwise, you will have duplicate IP addresses in the network.

2. Change the virtual router priorities.

Virtual routers 1–4 need to have their priority set to 100 from 101, and virtual routers 5–7 need to have their priorities set to 101 from 100. You can find this in line `/cfg/vrrp/vr 1/vrid 1/if 1/prio 101`.

3. **Scroll to the bottom of the text file and delete anything past “Script End.”**
4. **Save the changes to the text file as “Customer Name” Switch 2.**

Move your serial cable to the console port on the second switch. Any configuration on it needs to be deleted by resetting it to factory settings, using the following command:

```
>> Main# /boot/conf factory/reset
```

You can tell if the switch is at factory default when you log on because the switch will prompt you if you want to use the step-by-step configuration process. When it does, respond: “No.”

5. **In HyperTerminal, go to transfer/send text file and send the Switch 2 text file. The configuration will dump into the switch. Simply type apply, then save. When you can type characters in the terminal session again, reboot the switch (/boot/reset).**

VRRP-Based Hot-Standby Configuration

A hot-standby configuration allows all processes to failover to a backup switch if any type of failure should occur. The primary application for hot-standby redundancy is to avoid bridging loops when using the Spanning Tree Protocol (STP), IEEE 802.1d. VRRP-based hot-standby supports the default Spanning Tree only. It does not support multiple Spanning Trees.

Figure 13-10 shows a classic network topology, designed with redundancy in mind. This topology contains bridging loops that would require the use of STP. In the typical network, STP failover time is 45-50 seconds, much longer than the typical failover rate using VRRP only.

NOTE – To use hot-standby redundancy, peer switches must have an equal number of ports.

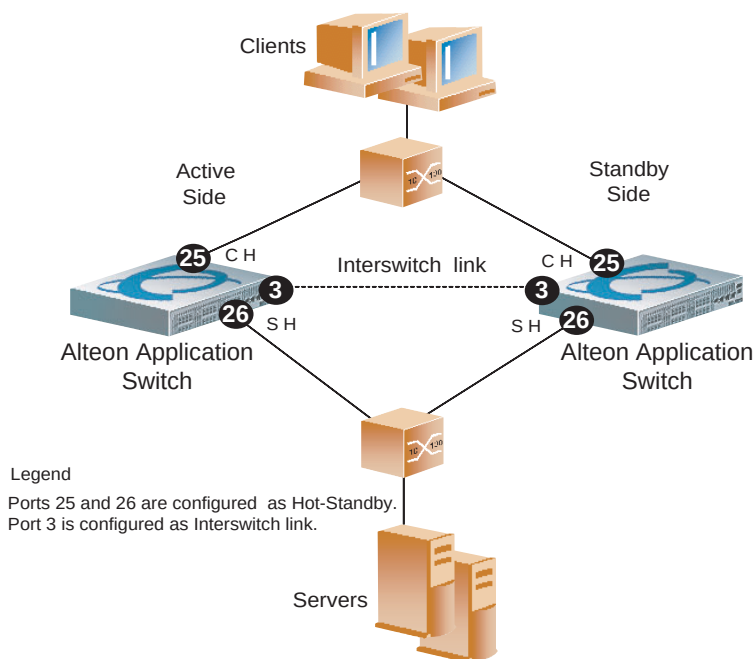


Figure 13-10 Hot-Standby Configuration

NOTE – In complex networks, STP convergence time can be much higher than 45-50 seconds.

If VRRP was used in this configuration, it would require STP. An important factor to consider is that the switch would be affected by the slower failover time of STP even if VRRP were in use.

While VRRP can be used without STP in this scenario, doing so would involve a more complex network configuration, requiring multiple subnets and/or VLANs and enabling IP forwarding to route between them.

By reducing complexity to a single subnet and not requiring routing (layer 3), hot-standby can be used. The key to hot-standby is that the *interswitch link* (the link between switches), does *not* participate in STP, so there are no loops in the topology (see [Figure 13-10](#)). STP does not need to be enabled, and the switch will have failover times similar to what would be the case with VRRP.

Configuration Procedure

Configuration takes place after configuring SLB and VRRP with STP enabled:

1. **From the SLB menu, enable a hot-standby link on the Layer 4 ports; then enable inter-switch link on the crosslink.**

```
>> Main# /cfg/slb/port 26           (Select port 26)
>> SLB Port 26# server ena          (Enable the server)
>> SLB Port 26# hotstan ena         (Enable hot standby)
>> SLB Port 26# ../port 3           (Select port 3)
>> SLB Port 3# intersw ena          (Enable inter-switch processing)
>> SLB Port 3# ../port 25           (Select port 25)
>> SLB Port 25# client ena          (Enable the client)
>> SLB Port 25# hotstan ena         (Enable hot standby)
```

2. **From the VRRP menu, enable VRRP group mode; then enable hot-standby.**

```
>> Main# /cfg/vrrp/on               (Enable VRRP)
>> VRRP# hotstan ena                (Enable hot standby)
>> VRRP# group ena                  (Enable VR group)
```

3. **Sync the VRRP, SLB, and filter settings to the other switch (same ports).**

```
>> Main# /oper/slb/sync
```

NOTE – Switches peering with each other must have an equal number of ports.

4. **Turn off STP after verifying that the network is stable.**

```
>> Main# /cfg/stp 1/off              (Disable STP group)
>> Spanning Tree Group 1# save       (Enable hot standby)
>> Main# /boot/reset                 (Reset the switch)
```

NOTE – You must reboot the switch for the hot-standby configuration to take effect.

Virtual Router Deployment Considerations

Review the following issues described in this section to prevent network problems when deploying virtual routers:

- [Mixing Active-Standby and Active-Active Virtual Routers](#)
- [Synchronizing Active/Active Failover](#)
- [Eliminating Loops with STP and VLANs](#)
- [Assigning VRRP Virtual Router ID](#)
- [Configuring the Switch for Tracking](#)
- [Synchronizing Configurations](#)

Mixing Active-Standby and Active-Active Virtual Routers

If the network environment can support sharing, enable it for all virtual routers in the LAN. If not, use active-standby for all virtual routers. Do not mix active-active and active-standby virtual routers in a LAN. Mixed configurations have not been tested, may result in unexpected operational characteristics, and, therefore, are not recommended.

Synchronizing Active/Active Failover

The hot-standby failover required the primary and secondary switches to have identical configurations and port topology. With VRRP and active/active failover, this is optional. Each switch can be configured individually with different port topology, SLB, and filters. If you would rather force two active/active switches to use identical settings, you can synchronize their configuration using the following command:

```
>> Main# /oper/slb/sync
```

The `sync` command copies the following settings to the switch at the specified IP interface address:

- VRRP settings
- SLB settings (including port settings)
- Filter settings (including filter port settings)
- Proxy IP settings

If you perform the `sync` command, you should check the configuration on the target switch to ensure that the settings are correct.

For more information on synchronizing configurations between two switches, see [“Synchronizing Configurations”](#) on page 321.

Eliminating Loops with STP and VLANs

VRRP active/active failover is significantly different from the hot-standby failover method supported in previous releases. As shown in [Figure 13-11](#), active-active configurations can introduce loops into complex LAN topologies.

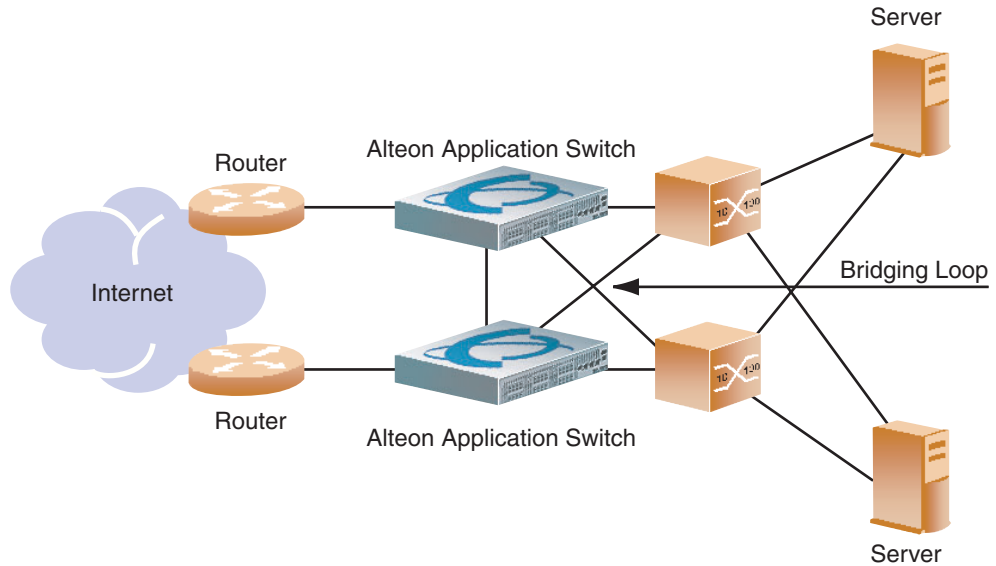


Figure 13-11 Loops in Active-Active Configuration

Using Spanning Tree Protocol to Eliminate Loops

VRRP generally requires Spanning Tree Protocol (STP) to be enabled in order to resolve bridge loops that usually occur in cross-redundant topologies, as shown in [Figure 13-12](#). In this example, a number of loops are wired into the topology. STP resolves loops by blocking ports where looping is detected.

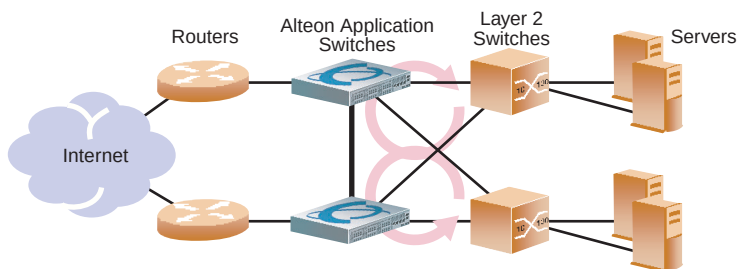


Figure 13-12 Cross-Redundancy Creates Loops, but STP Resolves Them

One drawback to using STP with VRRP is the failover response time. STP could take as long as 45 seconds to re-establish alternate routes after a switch or link failure.

Using VLANs to Eliminate Loops

When using VRRP, you can decrease failover response time by using VLANs instead of STP to separate traffic into non-looping broadcast domains. An example is shown in [Figure 13-13](#):

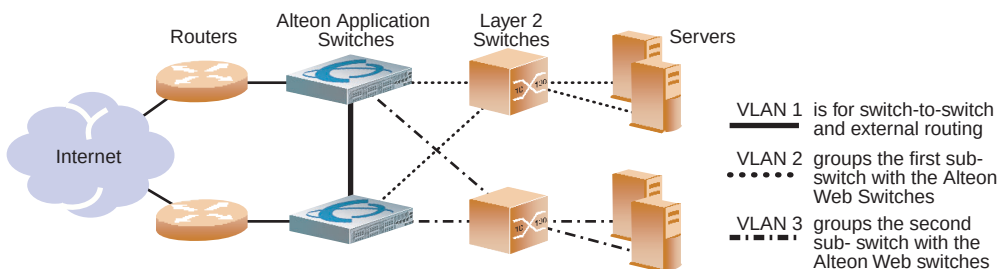


Figure 13-13 Using VLANs to Create Non-Looping Topologies

This topology allows STP to be disabled. On the Alteon Application Switches, IP routing allows traffic to cross VLAN boundaries. The servers use the Alteon Application Switches as default gateways. For port failure, traffic is rerouted to the alternate path within one health check interval (configurable between 1 and 60 seconds, with a default of 2 seconds).

Assigning VRRP Virtual Router ID

During the software upgrade process, VRRP virtual router IDs will be automatically assigned if failover is enabled on the switch. When configuring virtual routers at any point after upgrade, virtual router ID numbers (`/cfg/vrrp/vr #/vrid`) must be assigned. The virtual router ID may be configured as any number between 1 and 255.

Configuring the Switch for Tracking

Tracking configuration largely depends on user preferences and network environment. Consider the configuration shown in [Figure 13-8 on page 302](#). Assume the following behavior on the network:

- Application switch 1 is the master router upon initialization.
- If application switch 1 is the master and it has one fewer active servers than application switch 2, then application switch 1 remains the master.
This behavior is preferred because running one server down is less disruptive than bringing a new master online and severing all active connections in the process.
- If application switch 1 is the master and it has two or more active servers fewer than application switch 2, then application switch 2 becomes the master.
- If application switch 2 is the master, it remains the master even if servers are restored on application switch 1 such that it has one fewer or an equal number of servers.
- If application switch 2 is the master and it has one active server fewer than application switch 1, then application switch 1 becomes the master.

The user can implement this behavior by configuring the switch for tracking as follows:

1. **Set the priority for application switch 1 to the default value of 100.**
2. **Set the priority for application switch 2 to 96.**
3. **On both switches, enable tracking based on the number of virtual routers in master mode on the switch and set the value = 5.**
4. **On both switches, enable tracking based on the number of healthy real servers behind the VIP address. The VIP address is the same as the IP address of the virtual server router on the switch. Set the value = 6.**

Initially, application switch 1 ([Figure 13-8 on page 302](#)) will have a priority of 100 (base value) + 5 (initially it is the master) + 24 (4 active real servers * 6 per real server) = 129.

Application Switch 2 will have a priority of 96 (base value) + 24 (4 active real servers * 6 per real server) = 120.

If a server attached to application switch 1 fails, then application switch 1's priority will be reduced by 6 to 123. Since 123 is greater than 120 (application switch 2's priority), application switch 1 will remain the master.

If a second server attached to application switch 1 fails, then application switch 1's priority will be reduced by 6 more to 117. Since 117 is less than 120 (application switch 2's priority), then application switch 2 will become the Master. At this point, application switch 1's priority will fall by 5 more and application switch 2's will rise by 5 because the switches are tracking how many masters they are running. So, application switch 1's priority will settle out at 112 and application switch 2's priority at 125.

When both servers are restored to application switch 1, that switch's priority will rise by 12 (2 healthy real servers * 6 per healthy server) to 124. Since 124 is less than 125, application switch 2 will remain the master.

If, at this point, a server fails on application switch 2, its priority will fall by 6 to 119. Since 119 is less than 124, application switch 1 will become the Master. Its priority will settle out at 129 (since it's now the master) while application switch 2's priority will drop by 5 more to 114.

You can see from this example that the user's goals were met by the configured tracking parameters.

NOTE – There is no shortcut to setting tracking parameters. The goals must first be set and the outcomes of various configurations and scenarios analyzed to find settings that meet the goals.

Synchronizing Configurations

As noted above, each VRRP-capable switch is autonomous. Switches in a virtual router need not be identically configured. As a result, configurations cannot be synchronized automatically.

For user convenience, it is possible to synchronize a configuration from one VRRP-capable switch to another using the `/oper/slb/sync` command. However, care must be taken when using this command to avoid unexpected results. All server load balancing, port configurations, filter configurations, and VRRP parameters can be synchronized using the `/oper/slb/sync` command.

NOTE – Before you synchronize the configuration between two switches, a peer must be configured on each switch. Switches being synchronized must use the same administrator password.

Configure the two switches as peers to each other. From Switch 1, configure Switch 2 as a peer and specify its IP address as follows:

>> Main # /cfg/slb/sync	<i>(Select the synchronization menu)</i>
>> Config Synchronization # peer 1	<i>(Select a peer)</i>
>> Peer Switch 1 # addr <IP address>	<i>(Assign switch 2 IP address)</i>
>> Peer Switch 1 # enable	<i>(Enable peer switch)</i>

Similarly, from Switch 2, configure Switch 1 as a peer and specify its IP address as follows:

>> Main # /cfg/slb/sync	<i>(Select the synchronization menu)</i>
>> Config Synchronization # peer 2	<i>(Select a peer)</i>
>> Peer Switch 2 # addr <IP address>	<i>(Assign switch 1 IP address)</i>
>> Peer Switch 2 # enable	<i>(Enable peer switch)</i>

Port specific parameters, such as what filters are applied and enabled on what ports, are part of what is pushed by the `/oper/slb/sync` command. Thus, if the `/oper/slb/sync` command is used, it is highly recommended that the hardware configurations and network connections of all switches in the virtual router be identical; that is, each switch should be the same model, have the same line cards in the same slots (if modular) and have the same ports connected to the same external network devices. Otherwise, unexpected results may occur when the `/oper/slb/sync` command attempts to configure a non-existent port or applies an inappropriate configuration to a port.

Stateful Failover of Layer 4 and Layer 7 Persistent Sessions

Alteon OS provides stateful failover of content-intelligent persistent session state and Layer 7 persistent session state. This includes the following:

- SSL session state
- HTTP cookie state
- Layer 4 persistent
- FTP session state
- WAP session state

Stateful failover is only supported in active-standby mode with VMA enabled. Providing stateful failover enables network administrators to mirror their Layer 7 and Layer 4 persistent transactional state on the peer switch.

NOTE – Stateful failover does not synchronize all sessions, except persistent sessions (SSL session ID persistence and cookie-based persistence). If a service fails in the middle of a connection, the current session is discarded, but the new connection binds the session request correctly to the same real server.

To provide stateful failover, the state of the connection and session table must be shared between the switches in high-availability configurations. If Virtual Matrix Architecture (VMA) is enabled, all URL and cookie-parsing information is stored in the session table on the last port number on the switch. Sharing this information between switches is necessary to ensure the persistent session goes back to the same server.

What Happens When a Switch Fails

Assume that the user performing an e-commerce transaction has selected a number of items and placed them in the shopping cart. The user has already established a persistent session on the top server in Figure 13-14. The user then clicks the **Submit** button to purchase the items. At this time, the active switch fails. With stateful failover, the following sequence of events occurs:

1. **The backup switch becomes active.**
2. **The incoming request is redirected to the backup switch.**
3. **When the user clicks Submit again, the request is forwarded to the correct server.**

Even though the master switch has failed, the stateful failover feature prevents the client from having to re-establish a secure session. The server that stores the secure session now returns a response to the client via the backup switch.

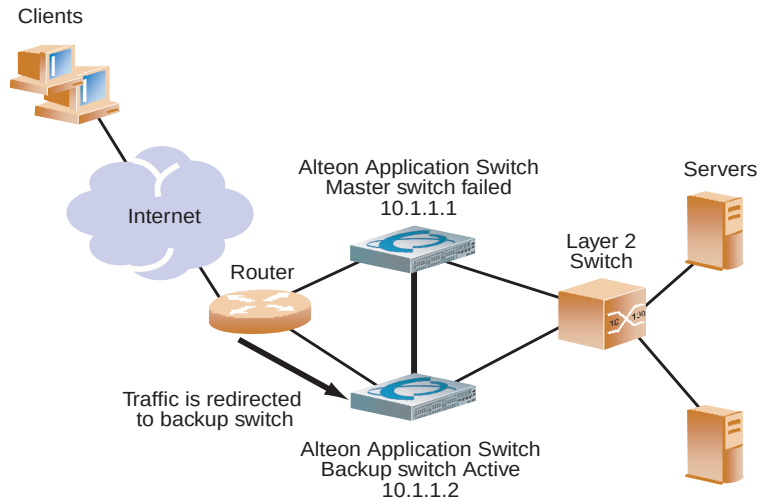


Figure 13-14 Stateful Failover Example when the Master Switch Fails

Stateful Failover Configuration Example

After the VRRP setup, perform the following additional steps to enable stateful failover on the switches.

On the Master Switch

1. Enable stateful failover.

```
>> # /cfg/slb/sync/state ena
```

2. Set the update interval.

```
>> # /cfg/slb/sync/update 10 (The default is 30)
```

3. Configure the backup switch as a peer and specify its IP address.

```
>> # /cfg/slb/sync
>> Config Synchronization # peer 1 (Select a peer)
>> Peer Switch 1 # addr 10.1.1.2 (Assign backup switch IP address)
>> Peer Switch 1 # enable (Enable peer switch)
```

On the Backup Switch

1. Turn on stateful failover.

```
>> # /cfg/slb/sync/state ena
```

2. Set the update interval.

```
>> # /cfg/slb/sync/update 10 (The default is 30)
```

NOTE – The update does not have to be the same for both switches. Stateful failover supports up to two peer switches. Repeat the steps mentioned above to enable stateful failover on all the peer switches.

3. Configure the master switch as a peer and specify its IP address.

```
>> # /cfg/slb/sync
>> Config Synchronization # peer 2           (Select a peer)
>> Peer Switch 2 # addr 10.1.1.1           (Assign master switch IP address)
>> Peer Switch 2 # enable                   (Enable peer switch)
```

Viewing Statistics on Persistent Port Sessions

You can view statistics on persistent port sessions using the `/stats/slb/ssl` command. To determine which switch is the master and which is the backup, use the `/info/vrrp` command. The column on the far right displays the switch status.

If the switch is a master:

```
>> # /info/vrrp                               (View VRRP Information)
VRRP information:
  1: vrid    1, 172.21.16.187,    if  4, renter, prio 109, master,
server
  3: vrid    3, 192.168.1.30,     if  2, renter, prio 109, master
  5: vrid    5, 172.21.16.10,     if  4, renter, prio 109, master
```

If the switch is a backup:

```
>> # /info/vrrp                               (View VRRP Information)
VRRP information:
  1: vrid    1, 172.21.16.187,    if  1, renter, prio 104, backup,
server
  3: vrid    3, 192.168.1.30,     if  3, renter, prio 104, backup
  5: vrid    5, 172.21.16.10,     if  1, renter, prio 104, backup
```


Part 4: Advanced Switching

Alteon OS can parse requests and classify flows using URLs, host tags, and cookies so that each request can be isolated and treated intelligently. This section describes the following advanced switching applications:

- Global Server Load Balancing
- Firewall Load Balancing
- Virtual Private Network Load Balancing
- Content Intelligent Switching
- Persistence
- Bandwidth Management



CHAPTER 14

Global Server Load Balancing

This chapter provides information for configuring Global Server Load Balancing (GSLB) across multiple geographic sites. The following topics are covered:

- “GSLB Overview” on page 330
- “Enabling GSLB on the Switch” on page 332
- “Configuring GSLB” on page 333
- “Proxy IP for Non-HTTP Redirects” on page 344
- “Verifying GSLB Operation” on page 348
- “Configuring GSLB Client Proximity Tables” on page 348
- “Using Border Gateway Protocol for GSLB” on page 351

GSLB Overview

GSLB allows balancing server traffic load across multiple physical sites. The Alteon GSLB implementation takes into account an individual site's health, response time, and geographic location to smoothly integrate the resources of the dispersed server sites for complete global performance.

Benefits

GSLB meets the following demands for distributed network services:

- High content availability is achieved through distributed content and distributed decision making. If one site becomes disabled, the others become aware of it and take up the load.
- There is no latency during client connection set up. Instant site hand-off decisions can be made by any distributed switch.
- The best performing sites receive a majority of traffic over a given period of time but are not overwhelmed.
- Switches at different sites regularly exchange information through the Distributed Site State Protocol (DSSP), and can trigger exchanges when any site's health status changes. This ensures that each active site has valid state knowledge and statistics.
- GSLB implementation takes geography as well as network topology into account.
- Creative control is given to the network administrator or Webmaster to build and control content by user, location, target application, and more.
- GSLB is easy to deploy, manage, and scale. Switch configuration is straightforward. There are no complex system topologies involving routers, protocols, and so forth.
- Flexible design options are provided.
- All IP protocols are supported.

How GSLB Works

GSLB is based on the Domain Name System (DNS) and proximity by source IP address. In the example in [Figure 14-1](#), a client is using a Web browser to view the Web site for the Foo Corporation at “www.foo corp.com.” The Foo Corporation has two Web sites: one in California, and one in Denver, each with identical content and available services. Both Web sites have an Alteon Application Switch configured for GSLB. These switches are also configured as the Authoritative Name Servers for “www.foo corp.com.”

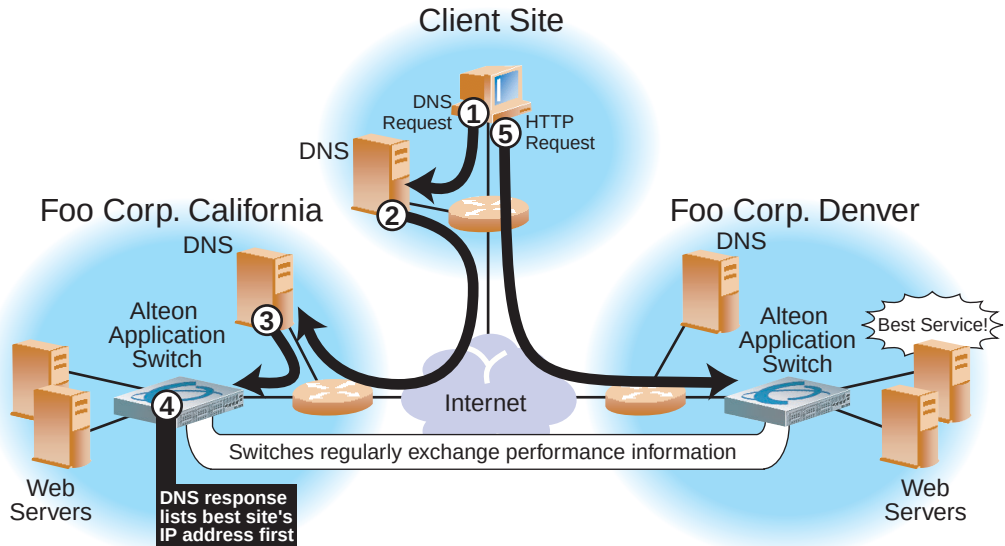


Figure 14-1 DNS Resolution with Global Server Load Balancing

The DNS resolution for GSLB is described in detail in the following procedure:

1. The client Web browser requests the “www.foo corp.com” IP address from the local DNS.
2. The client’s DNS asks its upstream DNS, which in turn asks the next, and so on, until the address is resolved.

Eventually, the request reaches an upstream DNS server that has the IP address information available or the request reaches one of the Foo Corporation’s DNS servers.

3. The Foo Corporation’s California DNS has been configured to use the local Alteon Application Switch with GSLB software as the authoritative name server for “www.foo corp.com.”

4. The California switch responds to the DNS request, listing the IP address with the current best service.

Each switch with GSLB software is capable of responding to the client's name resolution request. Since each switch regularly checks and communicates health and performance information with its peers, either switch can determine which site(s) are best able to serve the client's Web access needs. It can respond with a list of IP addresses for the Foo Corporation's distributed sites, which are prioritized by performance, geography, and other criteria.

In this case, the California switch knows that Foo Corp. Denver currently provides better service, and lists Foo Corp. Denver's virtual server IP address first when responding to the DNS request.

5. The client connects to Foo Corp. Denver for the best service.

The client's Web browser will use the IP address information obtained from the DNS request to open a connection to the best available site. The IP addresses represent virtual servers at any site, which are locally load balanced according to regular SLB configuration.

If the site serving the client HTTP content suddenly experiences a failure (no healthy real servers) or becomes overloaded with traffic (all real servers reach their maximum connection limit), the switch issues an HTTP Redirect and transparently causes the client to connect to another peer site.

The end result is that the client gets quick, reliable service with no latency and no special client-side configuration.

Enabling GSLB on the Switch

To use GSLB on your application switch, you must purchase an additional software license and password key. Instructions for obtaining a password key are provided with your software license certificate.

Once you have obtained the proper password key to enable GSLB, follow these instructions:

Connect to the switch command line interface via Telnet or the console port, and login as the administrator, following the directions in the *Alteon OS Command Reference*.

From the command line prompt, type the `/oper/swkey` command. You will be prompted to enter the password key. If the key is correct for this MAC address, the switch will accept the password, permanently record it in the switch's non-volatile RAM (NVRAM), and will then enable the feature.

Configuring GSLB

Configuring GSLB is simply an extension of the configuration procedure for SLB. The process is summarized as follows:

- Use the administrator login to connect to the switch you want to configure.
- Activate SLB and GSLB software keys. See the *Alteon OS Command Reference* for details.
- Configure the switch at each site with basic attributes.
 - Configure the switch IP interface.
 - Configure the default gateways.
- Configure the switch at each site to act as the DNS server for each service that is hosted on its virtual servers. Also, configure the local DNS server to recognize the switch as the authoritative DNS server for the hosted services.
- Configure the switch at each site for local SLB.
 - Define each local real server.
 - Group local real servers into real server groups.
 - Define the local virtual server with its IP address, services, and real server groups.
 - Define the switch port states.
 - Enable SLB.
- Finally, configure each switch so that it recognizes its remote peers.
 - Configure a remote real server entry on each switch for each remote service.
 - Add the remote real server entry to an appropriate real server group.
 - Enable GSLB.

Example GSLB Topology

Consider the following example network:

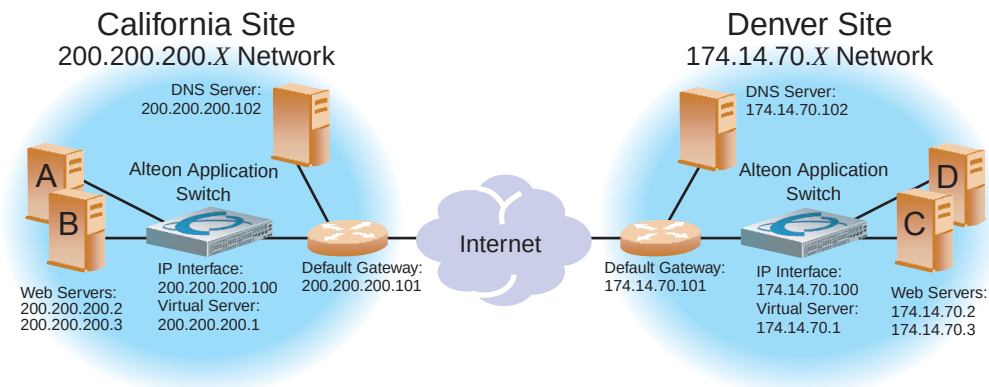


Figure 14-2 GSLB Topology Example

In the following examples, many of the options are left to their default values. See [“Additional Server Load Balancing Options” on page 148](#) for more options.

GSLB Requirements

The following is required prior to configuration:

- You must be connected to the switch Command Line Interface (CLI) as the administrator.
- Both of the following optional software keys must be activated:
 - ☐ SLB
 - ☐ GSLB

NOTE – For details about any of the processes or menu commands described in this example, see the *Alteon OS Command Reference*.

Task 1: Configure the Basics at the California Site

1. **If the Browser-Based Interface (BBI) is to be used for managing the California switch, change its service port.**

GSLB uses service port 80 on the IP interface for DSSP updates. By default, the Alteon OS Browser-Based Interface (BBI) also uses port 80. Both services cannot use the same port. If the BBI is enabled (see the `/cfg/sys/http` command in the *Alteon OS Command Reference*), configure it to use a different port.

For example, enter the following command to change the BBI port to 8080:

>> # <code>/cfg/sys</code>	<i>(Select the System menu)</i>
>> System# <code>wport 8080</code>	<i>(Set service port 8080 for BBI)</i>

2. **On the California switch, define an IP interface.**

The switch IP interface responds when asked to resolve client DNS requests. The IP interface must have an IP route to the local real servers. The switch uses this path to determine if the real servers can be reached via TCP/IP.

To configure an IP interface for this example, enter these commands from the CLI:

>> System# <code>/cfg/ip/if 1</code>	<i>(Select IP interface 1)</i>
>> IP Interface 1# <code>addr 200.200.200.100</code>	<i>(Assign IP address for the interface)</i>
>> IP Interface 1# <code>ena</code>	<i>(Enable IP interface 1)</i>

NOTE – This example assumes that all ports and IP interfaces use default VLAN 1, requiring no special VLAN configuration for the ports or IP interface.

3. **On the California switch, define the default gateway.**

The router at the edge of the site acts as the default gateway to the Internet. To configure the default gateway for this example, enter these commands from the CLI:

>> IP Interface 1# <code>./gw 1</code>	<i>(Select default gateway 1)</i>
>> Default gateway 1# <code>addr 200.200.200.101</code>	<i>(Assign IP address for the gateway)</i>
>> Default gateway 1# <code>ena</code>	<i>(Enable default gateway 1)</i>

4. **Configure the local DNS server to recognize the local GSLB switch as the authoritative name server for the hosted services.**

Determine the domain name that will be distributed to both sites and the host name for each distributed service. In this example, the California DNS server is configured to recognize 200.200.200.100 (the IP interface of the California GSLB switch) as the authoritative name server for “www.foocorp.com.”

Task 2: Configure the California Switch for Standard SLB

1. Assign an IP address to each of the real servers in the local California server pool.

The real servers in any real server group must have an IP route to the switch that will perform the SLB functions. This is most easily accomplished by placing the switches and servers on the same IP subnet, although advanced routing techniques can be used as long as they do not violate the topology rules outlined in [“Network Topology Requirements” on page 142](#).

For this example, the host real servers have IP addresses on the same IP subnet:

Table 14-1 GSLB Example: California Real Server IP Addresses

Real Server	IP address
Server A	200.200.200.2
Server B	200.200.200.3

2. On the California switch, define each local real server.

For each local real server, you must assign a real server number, specify its actual IP address, and enable the real server. For example:

```
>> Default gateway 1# /cfg/slb/real 1      (Server A is real server 1)
>> Real server 1# rip 200.200.200.2      (Assign IP address to server A)
>> Real server 1# ena                     (Enable real server 1)
>> Real server 1# ../real 2              (Server B is real server 2)
>> Real server 2# rip 200.200.200.3      (Assign IP address to server B)
>> Real server 2# ena                     (Enable real server 2)
```

3. On the California switch, define a real server group.

Combine the real servers into one service group and set the necessary health checking parameters. In this example, HTTP health checking is used to ensure that Web content is being served. If the `index.html` file is not accessible on a real server during health checks, the real server will be marked as down.

```
>> Real server 2# ../group 1              (Select real server group 1)
>> Real server group 1# add 1             (Add real server 1 to group 1)
>> Real server group 1# add 2             (Add real server 2 to group 1)
>> Real server group 1# health http       (Use HTTP for health checks)
>> Real server group 1# content index.html (Set URL content for health checks)
```


4. On the California switch, define a virtual server.

All client requests will be addressed to a virtual server IP address defined on the switch. Clients acquire the virtual server IP address through normal DNS resolution. HTTP uses well-known TCP port 80. In this example, HTTP is configured as the only service running on this virtual server IP address and, is associated with the real server group. For example:

```
>> Real server group 1# ../virt 1           (Select virtual server 1)
>> Virtual server 1# vip 200.200.200.1      (Assign a virtual server IP address)
>> Virtual Server 1# service 80
>> Virtual server 1 http Service# group 1   (Associate virtual port to real group)
>> Virtual server 1 http Service# ../ena    (Enable virtual server)
```

NOTE – This configuration is not limited to HTTP services. For a list of other well-known TCP/IP services and ports, see [Table 9-3 on page 148](#).

5. On the California switch, define the type of Layer 4 traffic processing each port must support.

In this example, the following ports are being used on the Alteon Application Switch:

Table 14-2 GSLB Example: California Alteon Switch Port Usage

Port	Host	Layer 4 Processing
1	Server A	Server
2	Server B	Server
6	Default Gateway Router. This connects the switch to the Internet where all client requests originate.	Client

The ports are configured as follows:

```
>> Virtual server 1# /cfg/slb/port 1        (Select physical switch port 1)
>> SLB port 1# server ena                  (Enable server processing on port 1)
>> SLB port 1# ../port 2                   (Select physical switch port 2)
>> SLB port 2# server ena                  (Enable server processing on port 2)
>> SLB port 2# ../port 6                   (Select physical switch port 6)
>> SLB port 6# client ena                  (Enable client processing on port 6)
```

6. On the California switch, enable SLB.

```
>> SLB port 6# /cfg/slb                    (Select the SLB Menu)
>> Layer 4# on                             (Turn SLB on)
```

Task 3: Configure the California Site for GSLB

1. On the California switch, define each remote site.

When you start configuring at the California site, California is local and Denver is remote. Add and enable the remote switch's IP address interface. In this example, there is only one remote site: Denver, with an IP interface address of 174.14.70.100. The following commands are used:

>> Layer 4# gslb/site 1	<i>(Select remote site 1)</i>
>> Remote site 1# prima 174.14.70.100	<i>(Define remote interface)</i>
>> Remote site 1# ena	<i>(Enable remote site 1)</i>

Each additional remote site would be configured in the same manner. You can enable up to 64 remote sites with a total aggregate of 2048 service/site combinations.

2. On the California switch, assign each remote distributed service to a local virtual server.

Configure the local California site to recognize the services offered at the remote Denver site. To do this, configure one real server entry on the California switch for each virtual server located at each remote site. Since there is only one remote site (Denver) with only one virtual server, only one more local real server entry is needed at the California site.

The new real server entry is configured with the IP address of the remote virtual server rather than the usual IP address of a local physical server. Do not confuse this value with the IP interface address on the remote switch.

Also, the *remote* parameter is enabled, and the real server entry is added to the real server group under the local virtual server for the intended service. Finally, since the real server health checks are performed across the Internet, the health-checking interval should be increased to 30 or 60 seconds to avoid generating excess traffic. For example:

>> Remote site 1# /cfg/slb/real 3	<i>(Create an entry for real server 3)</i>
>> Real server 3# rip 174.14.70.1	<i>(Set remote virtual server IP address)</i>
>> Real server 3# remote enable	<i>(Define the real server as remote)</i>
>> Real server 3# inter 60	<i>(Set a high health check interval)</i>
>> Real server 3# ena	<i>(Enable the real server entry)</i>
>> Real server 3# ../group 1	<i>(Select appropriate real server group)</i>
>> Real server group 1# add 3	<i>(Add real server 3 to the group 1)</i>

NOTE – Take care to note where each configured value originates or this step can result in improper configuration.

3. On the California switch, define the domain name and host name for each service hosted on each virtual server.

In this example, the domain name for the Foo Corporation is “foocorp.com,” and the host name for the only service (HTTP) is “www.” These values are configured as follows:

```
>> Real server group 1# ../virt 1          (Select virtual server 1)
>> Virtual server 1# dname foocorp.com      (Define domain name)
>> Virtual server 1# service 80/hname www   (Define HTTP host name)
```

To define other services (such as FTP), make additional hostname entries.

4. On the California switch, turn on GSLB.

```
>> Virtual server 1# ../gslb              (Select the GSLB Menu)
>> Global SLB# on                         (Activate GSLB for the switch)
```

5. Apply and verify the configuration.

```
>> Global SLB# apply                     (Make your changes active)
>> Global SLB# cur                       (View current GSLB settings)
>> Global SLB# /cfg/slb/cur             (View current SLB settings)
```

Examine the resulting information. If any settings are incorrect, make and apply any appropriate changes, and then check again.

6. Save your new configuration changes.

```
>> Layer 4# save                         (Save for restore after reboot)
```

Task 4: Configure the Basics at the Denver Site

Following the same procedure described for California (see [“Example GSLB Topology” on page 334](#)), configure the Denver site as follows:

1. If the Alteon OS BBI is to be used for managing the Denver switch, change its service port.

```
>> # /cfg/sys                           (Select the System menu)
>> System# wport 8080                   (Set service port 8080 for BBI)
```

2. On the Denver switch, define an IP interface.

```
>> # /cfg/ip/if 1 (Select IP interface 1)
>> IP Interface 1# addr 174.14.70.100 (Assign IP address for the interface)
>> IP Interface 1# ena (Enable IP interface 1)
```

3. On the Denver switch, define the default gateway.

```
>> IP Interface 1# ../gw 1 (Select default gateway 1)
>> Default gateway 1# addr 174.14.70.101 (Assign IP address for the gateway)
>> Default gateway 1# ena (Enable default gateway 1)
```

4. Configure the local DNS server to recognize the local GSLB switch as the authoritative name server for the hosted services.

The Denver DNS server is configured to recognize 174.14.70.100 (the IP interface of the Denver GSLB switch) as the authoritative name server for “www.foo corp.com.”

Task 5: Configure the Denver Switch for Standard SLB

1. Assign an IP address to each of the real servers in the local Denver server pool.

Table 14-3 Denver Real Server IP Addresses

Real Server	IP address
Server C	179.14.70.2
Server D	179.14.70.2

2. On the Denver switch, define each local real server.

```
>> Default gateway 1# /cfg/slb/real 1 (Server C is real server 1)
>> Real server 1# rip 179.14.70.2 (Assign IP address for Server C)
>> Real server 1# ena (Enable real server 1)
>> Real server 1# ../real 2 (Server D is real server 2)
>> Real server 2# rip 179.14.70.3 (Assign IP address for Server D)
>> Real server 2# ena (Enable real server 2)
```

3. On the Denver switch, define a real server group.

```
>> Real server 2# ../group 1           (Select real server group 1)
>> Real server group 1# add 1          (Add real server 1 to group 1)
>> Real server group 1# add 2          (Add real server 2 to group 1)
>> Real server group 1# health http    (Use HTTP for health checks)
>> Real server group 1# content index.html (Set URL content for health checks)
```

4. On the Denver switch, define a virtual server.

```
>> Real server group 1# ../virt 1      (Select virtual server 1)
>> Virtual server 1# vip 179.14.70.1   (Assign IP address)
>> Virtual server 1# service http      (Select the HTTP service menu)
>> Virtual server 1 http service# group 1 (Associate virtual port to real group)
>> Virtual server 1 http service# ../ena (Enable the virtual server)
```

5. On the Denver switch, define the type of Layer 4 processing each port must support.

In this example, the following ports are being used on the Alteon Application Switch:

Table 14-4 Web Host Example: Port Usage

Port	Host	Layer 4 Processing
3	Server C	Server
4	Server D	Server
5	Default Gateway Router. This connects the switch to the Internet where all client requests originate.	Client

The ports are configured as follows:

```
>> Virtual server 1# /cfg/slb/port 3    (Select physical switch port 3)
>> SLB port 3# server ena              (Enable server processing on port 3)
>> SLB port 3# ../port 4              (Select physical switch port 4)
>> SLB port 4# server ena              (Enable server processing on port 4)
>> SLB port 4# ../port 5              (Select physical switch port 5)
>> SLB port 5# client ena              (Enable client processing on port 5)
```

6. On the Denver switch, enable SLB.

```
>> SLB port 5# /cfg/slb                (Select the SLB Menu)
>> Layer 4# on                         (Turn SLB on)
```

Task 6: Configure the Denver Site for GSLB

Following the same procedure described for California (see [“Task 3: Configure the California Site for GSLB” on page 338](#)), configure the Denver site as follows:

1. On the Denver switch, define each remote site.

When you start configuring at the Denver site, Denver is local and California is remote. Add and enable the remote switch’s IP address interface. In this example, there is only one remote site: California, with an IP interface address of 200.200.200.100. The following commands are used:

>> Layer 4# gslb/site 1	<i>(Select remote site 1)</i>
>> Remote site 1# prima 200.200.200.100	<i>(Define remote IP interface address)</i>
>> Remote site 1# ena	<i>(Enable remote site 1)</i>

Each additional remote site would be configured in the same manner. You can enable up to 64 remote sites with a total aggregate of 2048 service/site combinations.

2. On the Denver switch, assign each remote distributed service to a local virtual server.

In this step, the local Denver site is configured to recognize the services offered at the remote California site. As before, configure one real server entry on the Denver switch for each virtual server located at each remote site. Since there is only one remote site (California) with only one virtual server, only one more local real server entry is needed at the Denver site.

The new real server entry is configured with the IP address of the remote virtual server, rather than the usual IP address of a local physical server. Do not confuse this value with the IP interface address on the remote switch.

Also, the *remote* parameter is enabled, and the real server entry is added to the real server group under the local virtual server for the intended service. Finally, since the real server health checks are headed across the Internet, the health-checking interval should be increased to 30 or 60 seconds to avoid generating excess traffic.

For example:

```
>> Remote site 1# /cfg/slb/real 3           (Create an entry for real server 3)
>> Real server 3# rip 200.200.200.1        (Set remote virtual server IP address)
>> Real server 3# remote enable            (Define the real server as remote)
>> Real server 3# inter 60                 (Set a high health check interval)
>> Real server 3# ena                      (Enable the real server entry)
>> Real server 3# ../group 1               (Select appropriate, real server group)
>> Real server group 1# add 3              (Add real server 3 to group 1)
```

NOTE – Take care to note where each configured value originates or this step can result in improper configuration.

3. On the Denver switch, define the domain name and host name for each service hosted on each virtual server.

These will be the same as for the California switch: the domain name is “foocorp.com,” and the host name for the HTTP service is “www.” These values are configured as follows:

```
>> Real server group 1# ../virt 1          (Select virtual server 1)
>> Virtual server 1# dname foocorp.com     (Define domain name)
>> Virtual server 1# service 80            (Select HTTP for virtual server)
>> Virtual server 1 http# hname www        (Define HTTP hostname)
```

4. On the Denver switch, turn on GSLB.

```
>> Virtual server 1 http# /cfg/slb/gslb/on (Activate GSLB for the switch)
```

5. Apply and verify the configuration.

```
>> Global SLB# apply                       (Make your changes active)
>> Global SLB# cur                         (View current GSLB settings)
>> Global SLB# /cfg/slb/cur               (View current SLB settings)
```

Examine the resulting information. If any settings are incorrect, make and apply any appropriate changes, and then check again.

6. Save your new configuration changes.

```
>> Layer 4# save                           (Save for restore after reboot)
```

Proxy IP for Non-HTTP Redirects

Typically, client requests for HTTP applications are automatically redirected to the location with the best response and least load for the requested content. This is because the HTTP protocol has a built-in redirection function that allows requests to be redirected to an alternate site. However, if a client requests a non-HTTP application such as FTP, POP3, or SMTP, then the lack of a redirection function in these applications requires that a proxy IP address be configured on the client port. The client port will initiate a redirect only if resources are unavailable at the first site.

NOTE – This feature should be used as a method of last resort for GSLB implementations in topologies where the remote servers are usually virtual server IP addresses in other Alteon Application Switches.

Figure 14-3 illustrates the packet-flow of HTTP and non-HTTP redirects in a GSLB environment. Table 14-5 explains the packet -flow process in detail. In this example, the initial DNS request from the client reaches Site 2, but Site 2 has no available services.

Table 14-5 HTTP Versus Non-HTTP Redirects

	Site 2 Alteon Switch	Site 1 Alteon Switch
HTTP application (built-in redirection)	1a. Client DNS request reaches Site 2. Resources are unavailable at Site 2. Site 2 sends a response to Client with Site 1’s virtual server IP address.	1b. Client resends request to Site 1. Resources are available at Site 1. Site 1 completes TCP three-way handshake with client.
Non-HTTP application (no redirection)	2a. Client DNS request reaches Site 2. Resources are unavailable at Site 2. Site 2 sends a request to Site 1 with Site 2’s proxy IP address as the source IP address and the virtual server IP address of Site 1 as the destination IP address.	2b. Site 1 processes the client proxy IP request. Resources are available at Site 1. Site 1 returns request to proxy IP port on Site 2. Site 2 completes the three-way handshake with Client.

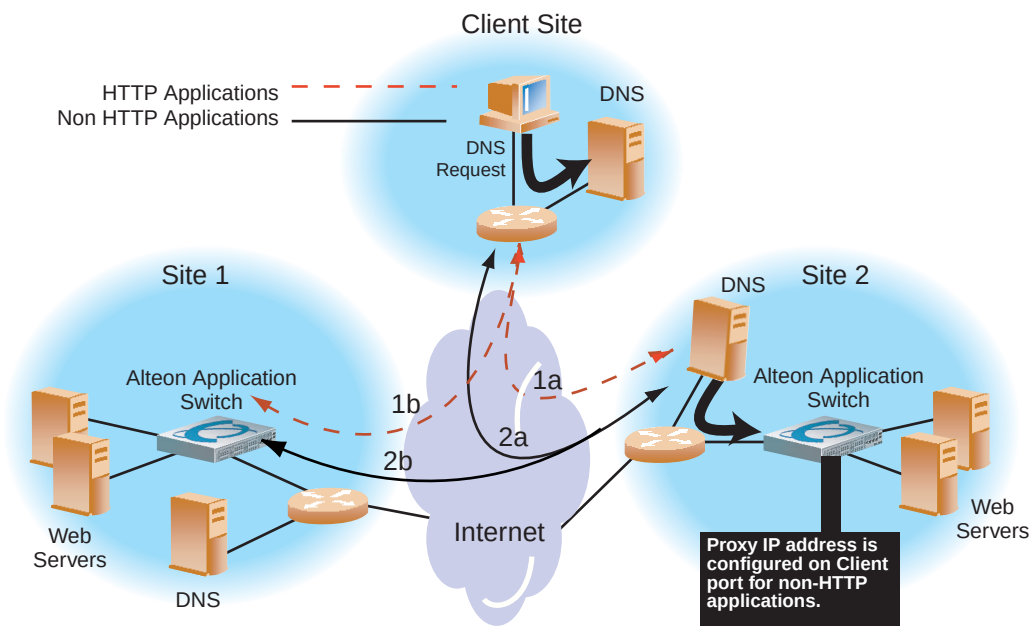


Figure 14-3 HTTP and Non-HTTP Redirects

How Proxy IP Works

Figure 14-4 shows examples of two GSLB sites deployed in California and Denver. The applications being load balanced are HTTP and POP3. Any request that cannot be serviced locally is sent to the peer site. HTTP requests are sent to the peer site using HTTP Redirect. Any other application request will be sent to the peer site using the proxy IP feature.

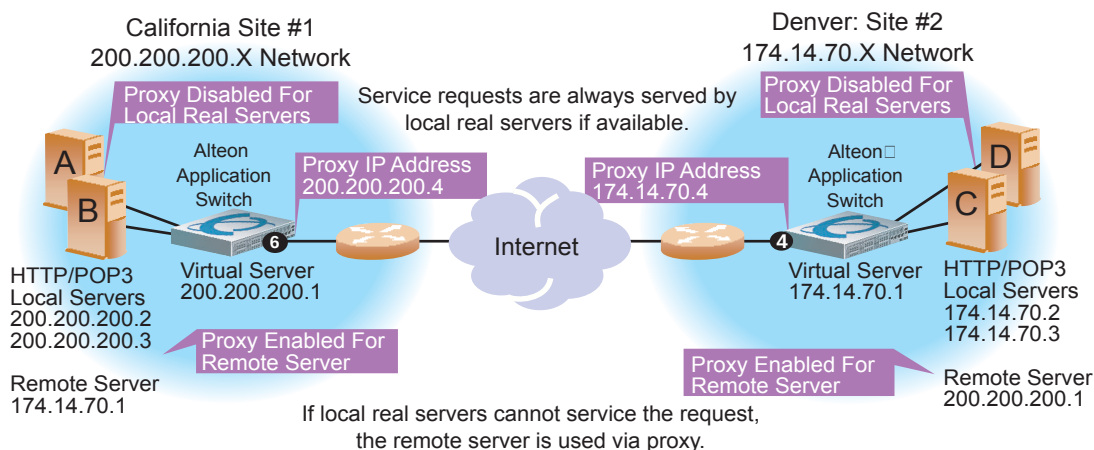


Figure 14-4 POP3 Request Fulfilled via IP Proxy

The following procedure explains the three-way handshake between the two sites and the client for a non-HTTP application (POP3). When POP3 processes at Site 1 terminate because of operator error, the following events occur to allow POP3 requests to be fulfilled:

1. A user POP3 TCP SYN request is received by the virtual server at Site 1. The switch at that site determines that there are no local resources to handle the request.
2. The Site 1 switch rewrites the request such that it now contains a client proxy IP address as the source IP address, and the virtual server IP address at Site 2 as the destination IP address.
3. The switch at Site 2 receives the POP3 TCP SYN request to its virtual server. The request looks like a normal SYN frame, so it performs normal local load-balancing.
4. Internally at Site 2, the switch and the real servers exchange information. The TCP SYN ACK from Site 2's local real server is sent back to the IP address specified by the proxy IP address.
5. The Site 2 switch sends the TCP SYN ACK frame to Site 1, with Site 2's virtual server IP address as the source IP address, and Site 1's proxy IP address as the destination IP address.

6. The switch at Site 1 receives the frame and translates it, using Site 1's virtual server IP address as the source IP address and the client's IP address as the destination IP address.

This cycle continues for the remaining frames to transmit all the client's mail, until a FIN frame is received.

Configuring Proxy IP Addresses

Refer to the example starting on [page 334](#) and [Figure 14-4](#), the switch at Site 1 in California is configured with switch port 6 connecting to the default gateway and real server 3 represents the remote server in Denver.

The following commands are used at Site 1 in California to configure the proxy IP address for the remote server in Denver:

NOTE – All four proxy IP addresses must be configured prior to enabling Virtual Matrix Architecture (VMA). Once proxy IP addresses are configured, ports not using proxy IP addresses can be disabled. For more information on proxy IP addresses, see [“Proxy IP Address and VMA” on page 157](#).

```
>> # /cfg/slb/pip                               (Select proxy IP address menu)
>> Proxy IP address# pip1 200.200.200.4         (Set unique proxy IP address)
>> # /cfg/slb/port 6/proxy enable                (Enable proxy on the port)
>> Proxy IP address ../real 1/proxy disable      (Disable local real server proxy)
>> Real server 1# ../real 2/proxy disable        (Disable proxy for local server)
>> Real server 2# ../real 3/proxy enable         (Enable proxy for remote server)
>> Real server 3# apply                          (Apply configuration changes)
>> Real server 3# save                          (Save configuration changes)
```

For more information on proxy IP address, see [“Proxy IP Addresses” on page 156](#).

If you want to configure proxy IP addresses on Site 2, the following commands are issued on the Denver switch:

```
>> # /cfg/slb/pip                               (Select proxy IP address menu)
>> Proxy IP address# pip1 174.14.70.4           (Set unique proxy IP address)
>> # /cfg/slb/port 4/proxy enable                (Enable proxy on the port)
>> Proxy IP address# ../real 1/proxy disable      (Disable local real server proxy)
>> Real server 1# ../real 2/proxy disable        (Disable local real server proxy)
>> Real server 2# ../real 3/proxy enable         (Enable proxy for remote server)
>> Real server 3# apply                          (Apply configuration changes)
>> Real server 3# save                          (Save configuration changes)
```

Verifying GSLB Operation

- Use your browser to request the configured service (`www.fooocorp.com` in the previous example).
- Examine the `/info/slb` information on each switch.
- Check to see that all SLB parameters are working according to expectation. If necessary, make any appropriate configuration changes and then check the information again.
- Examine the following statistics on each switch:
 - `/stats/slb/gslb/virt` *<virtual server number>*
 - `/stats/slb/gslb/group` *<real server group number>*
 - `/stats/slb/maint`

Configuring GSLB Client Proximity Tables

Alteon OS software allows clients to select GSLB sites based on where the client is located. This is implemented by configuring static “client-to-site” mapping called client proximity tables.

The use of a static client/site database allows you to customize the client environment. The proximity database is limited to 128 entries. The GSLB client proximity table is supported for HTTP protocol.

Figure 14-5 illustrates GSLB proximity tables. The client sends a request to the DNS server, which is forwarded to the master switch. The master switch looks through its proximity table and returns the request to the DNS server with the virtual server IP address of the preferred site. The DNS server sends a new request through the Internet and connects the client to the preferred site.

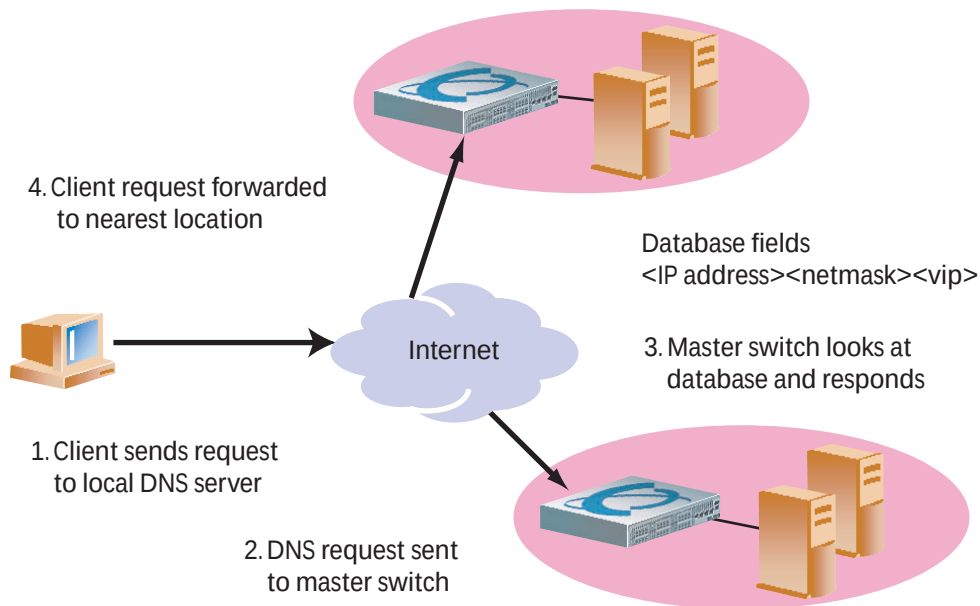


Figure 14-5 GSLB Proximity Tables: How They Work

The following example, illustrated in Figure 14-6 on page 350, shows how to add entries into a GSLB proximity table. Two client networks, A and B, are configured in the proximity table on the master switch at Site 4. Client A with a subnet address of 205.178.13.0 is configured with static entries for preferred Sites 1 and 3. Client B, with a subnet address of 204.165.0.0, is configured with static entries for preferred Sites 2 and 4.

Client A, with a source IP address of 205.178.13.10, initiates a request that is sent to the local DNS server. The local DNS server is configured to forward requests to the DNS server at Site 4. The Alteon Application Switch at Site 4 looks up its proximity table and finds Client A prefers to connect to Sites 1 or 3. Similarly, Client B's requests are always forwarded to Sites 2 or 4.

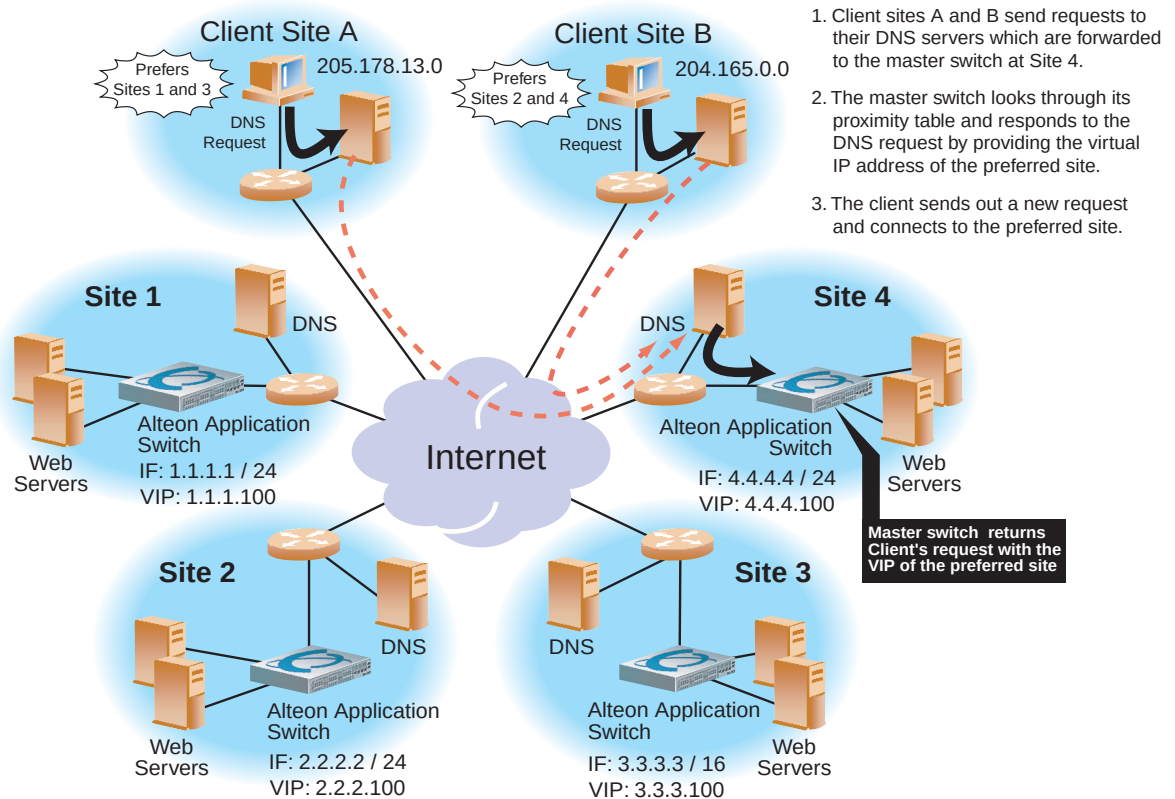


Figure 14-6 Configuring Client Proximity Table

You can add static entries in the proximity table for up to 128 client networks.

NOTE – Alteon OS allows you to configure a single domain only and for each client subnet you can configure up to two preferred sites.

The application switch forwards the client request based on the minimum available sessions and response time between the two preferred sites.

Use the following commands to configure a proximity table on the application switch at Site 4:

>> # /cfg/slb/gslb/lookup/lookups ena	(Enable the lookup or proximity table)
>> # dname nortelnetworks.com	(Select the domain name)
>> # network 1	(Select Client A subnet)
>> # sip 205.178.13.0	(Assign source address for Client A)
>> # mask 255.255.255.0	(Assign the mask for Client A)
>> # vip1 <virtual server IP addr of Site 3>	(Select preferred Site 3)
>> # vip2 <virtual server IP addr of Site 1>	(Select preferred Site 1)
>> # ../network 2	(Select Client B subnet)
>> # sip 204.165.0.0	(Assign source address for Client B)
>> # mask 255.255.0.0	(Assign the mask for Client B)
>> # vip1 <virtual server IP add of Site 4>	(Select preferred Site 4)
>> # vip2 <virtual server IP add of Site 2>	(Select preferred Site 2)

NOTE – For each client subnet, add only one static entry.

Using this configuration, the DNS request “nortelnetworks.com” from 205.178.13.0 will get a DNS response with only the virtual server IP address of Site 1, if Site 1 has less load than Site 3.

Using Border Gateway Protocol for GSLB

Border Gateway Protocol (BGP)-based GSLB utilizes the Internet’s routing protocols to localize content delivery to the most efficient and consistent site. This is done by using a shared IP block that co-exists in each Internet Service Provider’s (ISP’s) network and is then advertised, using BGP, throughout the Internet.

Because of the way IP routing works, BGP-based GSLB allows for the routing protocols to route DNS requests to the closest location, which then returns IP addresses of that particular site, locking in the requests to that site. In effect, the Internet is making the decision of the best location for you, avoiding the need for advanced GSLB.

Some corporations use more than one ISP as a way to increase the reliability and bandwidth of their Internet connection. Enterprises with more than one ISP are referred to as being *multi-homed*. Instead of multi-homing a network to several other networks, BGP-based GSLB enables you to multi-home a particular piece of content (DNS information) to the Internet by distributing the IP blocks that contain that content to several sites.

When using DNS to select the site, a single packet is used to make the decision so that the request will not be split to different locations. Through the response to the DNS packet, a client is locked in to a particular site, resulting in efficient, consistent service that cannot be achieved when the content itself is shared.

For example, in multi-homing, you can connect a data center to three different Internet providers and have one DNS server that has the IP address 1.1.1.1. In this case, all requests can be received via any given feed but are funneled to the same server on the local network. In BGP-based GSLB, the DNS server (with the IP address 1.1.1.1) is duplicated and placed local to the *peering point* instead of having a local network direct traffic to one server.

When a particular DNS server receives a request for a record (in this case, the application switch), it returns with the IP address of a virtual server at the same site. This can be achieved using the `local` option (`/cfg/slb/gslb/local ena`) in the GSLB configuration. If one site is saturated, then the switch will failover and deliver the IP address of a more available site.

For more information on configuring your switch to support BGP routing, see [“Border Gateway Protocol” on page 87](#).



CHAPTER 15

Firewall Load Balancing

Firewall Load Balancing (FWLB) with Alteon Application Switches allows multiple active firewalls to operate in parallel. Parallel operation allows users to maximize firewall productivity, scale firewall performance without forklift upgrades, and eliminate the firewall as a single point-of-failure.

This chapter presents the following topics:

■ **“Firewall Overview” on page 354**

An overview of firewalls and the various FWLB solutions supported by Alteon Application switches.

■ **“Basic FWLB” on page 356**

Explanation and example configuration for FWLB in simple networks, using two parallel firewalls and two application switches. The basic FWLB method combines redirection filters and static routing for FWLB.

■ **“Four-Subnet FWLB” on page 368**

Explanation and example configuration for FWLB in a large-scale, high-availability network with redundant firewalls and application switches. This method combines redirection filters, static routing, and Virtual Router Redundancy Protocol (VRRP).

■ **“Advanced FWLB Concepts” on page 389**

- **“Free-Metric FWLB” on page 389.** Using other load balancing metrics (besides hash) by enabling the Return to Sender (RTS) option.
- **“Adding a Demilitarized Zone (DMZ)” on page 393.** Adding a DMZ for servers that attach to the Alteon Application Switch between the Internet and the firewalls.
- **“Firewall Health Checks” on page 395.** Methods for fine-tuning the health checks performed for FWLB.

Firewall Overview

Firewall devices have become indispensable for protecting network resources from unauthorized access. Prior to FWLB, however, firewalls could become critical bottlenecks or single points-of-failure for your network.

As an example, consider the following network:

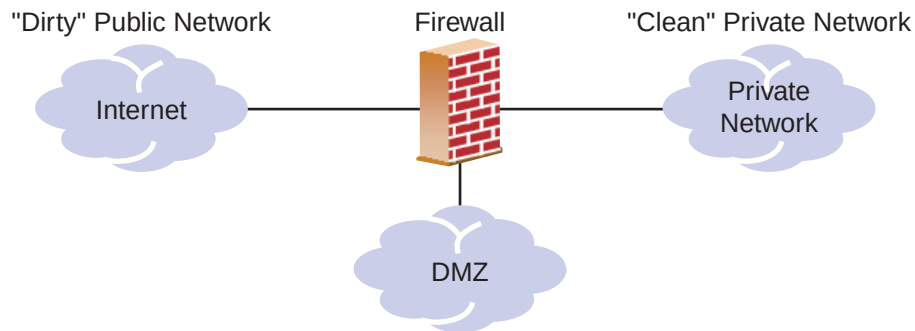


Figure 15-1 Typical Firewall Configuration Before FWLB

One network interface card on the firewall is connected to the public side of the network, often to an Internet router. This is known as the *dirty* or untrusted side of the firewall. Another network interface card on the firewall is connected to the side of the network with the resources that must be protected. This is known as the *clean* or trusted side of the firewall.

In this simple example, all traffic passing between the dirty, clean, and DMZ networks must traverse the firewall, which examines each individual packet. The firewall is configured with a detailed set of rules that determine which types of traffic are allowed and which types are denied. Heavy traffic can turn the firewall into a serious bottleneck. The firewall is also a single point-of-failure device. If it goes out of service, external clients can no longer reach your services and internal clients can no longer reach the Internet.

Sometimes, a *Demilitarized Zone* (DMZ) is attached to the firewall or between the Internet and the firewall. Typically, a DMZ contains its own servers that provide dirty-side clients with access to services, making it unnecessary for dirty-side traffic to use clean-side resources.

FWLB with Alteon Application Switches provides a variety of options that enhance firewall performance and resolve typical firewall problems.

Alteon Application Switches support the following methods of FWLB:

- Basic FWLB for simple networks

This method uses a combination of static routes and redirection filters and is usually employed in smaller networks.

An Alteon Application Switch filter on the dirty-side splits incoming traffic into streams headed for different firewalls. To ensure persistence of session traffic through the same firewall, distribution is based on a mathematical *hash* of the IP source and destination addresses.

For more information about basic FWLB, see [“Basic FWLB” on page 356](#).

- Four-Subnet FWLB for larger networks

Although similar to basic FWLB, the four-subnet method is more often deployed in larger networks that require high-availability solutions. This method adds Virtual Router Redundancy Protocol (VRRP) to the configuration.

Just as with the basic method, four-subnet FWLB uses the hash metric to distribute firewall traffic and maintain persistence.

For more information, see [“Four-Subnet FWLB” on page 368](#).

Each method is described in more detail in the following sections.

Basic FWLB

The basic FWLB method uses a combination of static routes and redirection filters to allow multiple active firewalls to operate in parallel.

Figure 15-2 shows a basic FWLB topology:

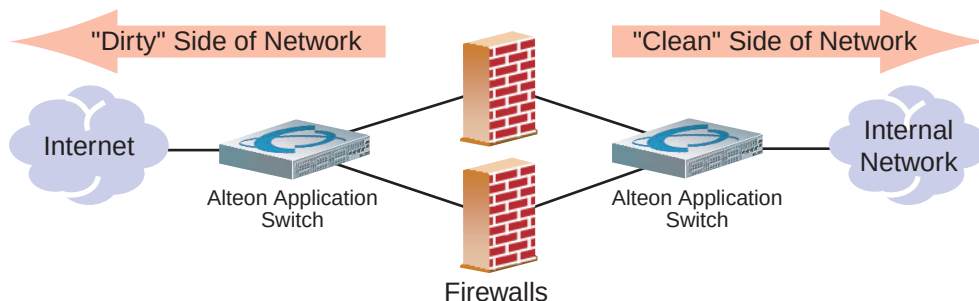


Figure 15-2 Basic FWLB Topology

The firewalls being load balanced are in the middle of the network, separating the dirty side from the clean side. This configuration requires a minimum of two application switches: one on the dirty side of the firewalls and one on the clean side.

A redirection filter on the dirty-side application switch splits incoming client traffic into multiple streams. Each stream is routed through a different firewall. The same process is used for outbound server responses; a redirection filter on the clean-side application switch splits the traffic, and static routes forward each stream through a different firewall and then back to the client.

Although other metrics can be used in some configurations (see [“Free-Metric FWLB” on page 389](#)), the distribution of traffic within each stream is normally based on a mathematical *hash* of the source IP address and destination IP addresses. This ensures that each client request and its related responses will use the same firewall (a feature known as *persistence*) and that the traffic will be equally distributed. The persistence is required for the firewall as it maintains state and processes traffic in both directions for a connection.

Although basic firewall load-balancing techniques can support more firewalls as well as multiple switches on the clean and dirty sides for redundancy, the configuration complexity increases dramatically. The four-subnet FWLB solution is usually preferred in larger scale, high-availability topologies (see [page 368](#)).

Basic FWLB Implementation

In this example, traffic is load balanced among the available firewalls.

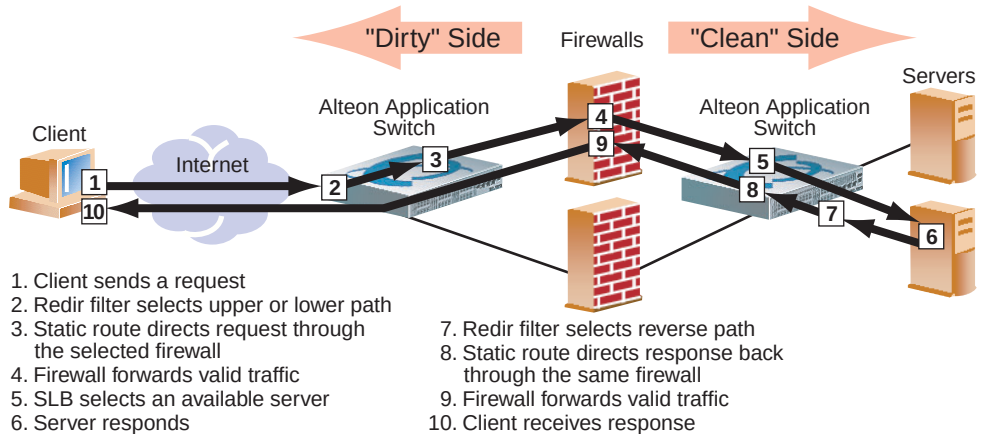


Figure 15-3 Basic FWLB Process

1. The client requests data.

The external clients intend to connect to services at the publicly advertised IP address assigned to a virtual server on the clean-side application switch.

2. A redirection filter balances incoming requests among different IP addresses.

When the client request arrives at the dirty-side application switch, a filter redirects it to a real server group that consists of a number of different IP addresses. This redirection filter splits the traffic into balanced streams: one for each IP address in the real server group. For FWLB, each IP address in the real server group represents an IP Interface (IF) on a different subnet on the clean-side application switch.

3. Requests are routed to the firewalls.

On the dirty-side switch, one static route is needed for each traffic stream. For instance, the first static route will lead to an IP interface on the clean-side application switch using the first firewall as the next hop. A second static route will lead to a second clean-side IP interface using the second firewall as the next hop, and so on. By combining the redirection filter and static routes, traffic is load balanced among all active firewalls.

All traffic between specific IP source/destination address pairs flows through the same firewall, ensuring that sessions established by the firewalls persist for their duration.

NOTE – More than one stream can be routed through a particular firewall. You can weight the load to favor one firewall by increasing the number of static routes that traverse it.

- 4. The firewalls decide if they should allow the packets and, if so, forwards them to a virtual server on the clean-side application switch.**

Client requests are forwarded or discarded according to rules configured for each firewall.

NOTE – Rule sets must be consistent across all firewalls.

- 5. The clean-side application switch performs normal SLB functions.**

Packets forwarded from the firewalls are sent to the original destination address, that is, the virtual server on the clean-side application switch. There, they are load balanced to the real servers using standard SLB configuration.

- 6. The real server responds to the client request.**

- 7. Redirection filters on the clean-side application switch balance responses among different IP addresses.**

Redirection filters are needed on all ports on the clean-side application switch that attach to real servers or internal clients on the clean-side of the network. Filters on these ports redirect the Internet-bound traffic to a real server group that consists of a number of different IP addresses. Each IP address represents an IP interface on a different subnet on the dirty-side application switch.

- 8. Outbound traffic is routed to the firewalls.**

Static routes are configured on the clean-side switch. One static route is needed for each stream that was configured on the dirty-side application switch. For instance, the first static route would be configured to lead to the first dirty-side IP interface using the first firewall as the next hop. The second static route would lead to the second dirty-side IP interface using the second firewall as the next hop, and so on.

Since application switches intelligently maintain state information, all traffic between specific IP source/destination addresses flows through the same firewall, maintaining session persistence.

NOTE – If Network Address Translation (NAT) software is used on the firewalls, FWLB session persistence requires the RTS option to be enabled on the application switch (see “Free-Metric FWLB” on page 389).

9. The firewall decides if it should allow the packet and, if so, forwards it to the dirty-side application switch.

Each firewall forwards or discards the server responses according to the rules that are configured for it. Forwarded packets are sent to the dirty-side application switch and out to the Internet.

10. The client receives the server response.

Configuring Basic FWLB

The steps for configuring basic FWLB are provided below. While two or four switches can be used, the following procedure assumes a simple network topology with only two application switches (one on each side of the firewalls) as shown in Figure 15-4.

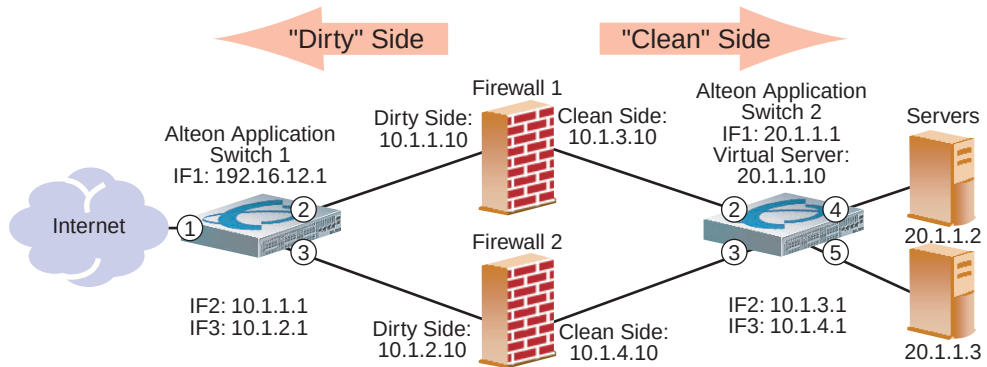


Figure 15-4 Basic FWLB Example Network

Configure the Dirty-Side Alteon Application Switch

1. Configure VLANs.

NOTE – Alternately, if using hubs between the switches and firewalls and you do not wish to configure VLANs, you must enable Spanning Tree Protocol to prevent broadcast loops.

2. Define the dirty-side IP interface.

In addition to one IP interface for general switch management, there must be one dirty-side IP interface for each firewall path being load balanced. Each must be on a different subnet.

```
>> # /cfg/ip/if 1 (Select IP interface 1)
>> IP Interface 1# addr 192.16.12.1 (Set address for switch management)
>> IP Interface 1# mask 255.255.255.0 (Set subnet mask for interface 1)
>> IP Interface 1# ena (Enable IP interface 1)
>> IP Interface 1# ../if 2 (Select IP interface 2)
>> IP Interface 2# addr 10.1.1.1 (Set the IP address for interface 2)
>> IP Interface 2# mask 255.255.255.0 (Set subnet mask for interface 2)
>> IP Interface 2# ena (Enable IP interface 2)
>> IP Interface 2# ../if 3 (Select IP interface 3)
>> IP Interface 3# addr 10.1.2.1 (Set the IP address for interface 3)
>> IP Interface 3# mask 255.255.255.0 (Set subnet mask for interface 3)
>> IP Interface 3# ena (Enable IP interface 3)
```

3. Configure the clean-side IP interface as if they were real servers on the dirty side.

Later in this procedure, you'll configure one clean-side IP interface on a different subnet for each firewall path being load balanced. On the dirty-side application switch, create two real servers using the IP address of each clean-side IP interface used for FWLB.

NOTE – The real server index number must be the same on both sides of the firewall. For example, if real server 1 is the dirty-side IP interface for Firewall 1, then configure real server 1 on the clean side with the dirty-side IP interface. Configuring the same real server number on both sides of the firewall ensures that the traffic travels through the same firewall.

```
>> IP Interface 3# /cfg/slb/real 1 (Select real server 1)
>> Real server 1# rip 10.1.3.1 (Assign clean-side IF 2 address)
>> Real server 1# ena (Enable real server 1)
>> Real server 1# ../real 2 (Select real server 2)
>> Real server 2# rip 10.1.4.1 (Assign clean-side IF 3 address)
>> Real server 2# ena (Enable real server 1)
```

Real servers in the server groups must be ordered the same on both clean side and dirty side switch. For example, if real server 1 IP interface connects to Firewall 1 for clean side server group, then real server 1 IP interface on the dirty side should be connected to Firewall 1. Selecting the same real server ensures that the traffic travels through the same firewall.

NOTE – Each of the four interfaces used for FWLB (two on each application switch) in this example must be configured for a different IP subnet.

4. Place the IP interface real servers into a real server group.

```
>> Real server 2# /cfg/slb/group 1      (Select real server group 1)
>> Real server group 1# add 1          (Add real server 1 to group 1)
>> Real server group 1# add 2          (Add real server 2 to group 1)
```

5. Set the health check type for the real server group to ICMP.

```
>> Real server group 1# health icmp      (Select ICMP as health check type)
```

6. Set the load-balancing metric for the real server group to hash.

```
>> Real server group 1# metric hash      (Select SLB hash metric for group 1)
```

Using the hash metric, all traffic between specific IP source/destination address pairs flows through the same firewall. This ensures that sessions established by the firewalls are maintained for their duration.

NOTE – Other load balancing metrics such as `leastconns`, `roundrobin`, `minmiss`, `response`, and `bandwidth` can be used when enabling the Return to Sender (RTS) option. For more information, see “Free-Metric FWLB” on page 389.

7. Enable SLB on the switch.

```
>> Real server group 1# /cfg/slb/on
```

8. Create a filter to allow local subnet traffic on the dirty side of the firewalls to reach the firewall interfaces.

```
>> Layer 4# /cfg/slb/filt 10            (Select filter 10)
>> Filter 10# sip any                   (From any source IP address)
>> Filter 10# dip 192.16.12.0           (Specify destination IP address)
>> Filter 10# dmask 255.255.255.0       (Specify destination mask)
>> Filter 10# action allow               (Allow frames with this DIP address)
>> Filter 10# ena                       (Enable filter)
```

9. Create the FWLB redirection filter.

This filter will redirect inbound traffic, load balancing it among the defined real servers in the group. In this network, the real servers represent IP interfaces on the clean-side application switch.

```
>> Filter 10# ../filt 15                (Select filter 15)
>> Filter 15# sip any                    (From any source IP address)
>> Filter 15# dip any                    (To any destination IP address)
>> Filter 15# proto any                  (For any protocol)
>> Filter 15# action redir               (Perform redirection)
>> Filter 15# group 1                   (To real server group 1)
>> Filter 15# ena                       (Enable the filter)
```

10. Enable firewall load balancing.

```
>> Filter 15# adv/fwlb ena
```

11. Add filters to the ingress port.

```
>> Filter 15# ../port 1                  (Select the ingress port)
>> SLB Port 1# add 10                    (Add the filter to the ingress port)
>> SLB Port 1# add 15                    (Add the filter to the ingress port)
>> SLB Port 1# filt ena                  (Enable filtering on the port)
```

12. Define static routes to the clean-side IP interfaces, using the firewalls as gateways.

One static route is required for each firewall path being load balanced. In this case, two paths are required: one that leads to clean-side IF 2 (10.1.3.1) through the first firewall (10.1.1.10) as its gateway, and one that leads to clean-side IF 3 (10.1.4.1) through the second firewall (10.1.2.10) as its gateway.

```
>> SLB Port 5# /cfg/ip/route
>> IP Static Route# add 10.1.3.1 255.255.255.255 10.1.1.10
>> IP Static Route# add 10.1.4.1 255.255.255.255 10.1.2.10
```

13. Apply and save the configuration changes.

```
>> # apply
>> # save
```

Configure the Clean-Side Alteon Application Switch

1. Define the clean-side IP interfaces.

Create one clean-side IP interface on a different subnet for each firewall being load balanced.

NOTE – An extra IP interface (IF 1) prevents server-to-server traffic from being redirected.

>> # /cfg/ip/if 1	(Select IP interface 1)
>> IP Interface 1# addr 20.1.1.1	(Set the IP address for interface 1)
>> IP Interface 1# mask 255.255.255.0	(Set subnet mask for interface 1)
>> IP Interface 1# ena	(Enable IP interface 1)
>> IP Interface 1# ../if 2	(Select IP interface 2)
>> IP Interface 2# addr 10.1.3.1	(Set the IP address for interface 2)
>> IP Interface 2# mask 255.255.255.0	(Set subnet mask for interface 2)
>> IP Interface 2# ena	(Enable IP interface 2)
>> IP Interface 2# ../if 3	(Select IP interface 3)
>> IP Interface 3# addr 10.1.4.1	(Set the IP address for interface 3)
>> IP Interface 3# mask 255.255.255.0	(Set subnet mask for interface 3)
>> IP Interface 3# ena	(Enable IP interface 3)

2. Configure the dirty-side IP interfaces as if they were real servers on the clean side.

You should already have configured a dirty-side IP interface on a different subnet for each firewall path being load balanced. Create two real servers on the clean-side switch, using the IP address of each dirty-side IP interface.

NOTE – The real server index number must be the same on both sides of the firewall. For example, if real server 1 is the dirty-side IP interface for Firewall 1, then configure real server 1 on the clean side with the dirty-side IP interface. Configuring the same real server number on both sides of the firewall ensures that the traffic travels through the same firewall.

>> IP Interface 5# /cfg/slb/real 1	(Select real server 1)
>> Real server 1# rip 10.1.1.1	(Assign dirty-side IF 1 address)
>> Real server 1# ena	(Enable real server 1)
>> Real server 1# ../real 2	(Select real server 2)
>> Real server 2# rip 10.1.2.1	(Assign dirty-side IF 2 address)
>> Real server 2# ena	(Enable real server 2)

NOTE – Each of the four IP interfaces (two on each application switch) in this example must be configured for a different IP subnet.

3. Place the real servers into a real server group.

>> Real server 2# ../group 1	<i>(Select real server group 1)</i>
>> Real server group 1# add 1	<i>(Select real server 1 to group 1)</i>
>> Real server group 1# add 2	<i>(Select real server 2 to group 1)</i>

4. Set the health check type for the real server group to ICMP.

>> Real server group 1# health icmp	<i>(Select ICMP as health check type)</i>

5. Set the load-balancing metric for the real server group to hash.

>> Real server group 1# metric hash	<i>(Select SLB hash metric for group 1)</i>

NOTE – The clean-side application switch must use the same metric as defined on the dirty side.

6. Enable server load balancing on the switch.

>> Real server group 1# /cfg/slb/on

7. Configure ports 2 and 3, which are connected to the clean-side of the firewalls, for client processing.

```
>> Real server group 1# ../port 2/client ena (Enable client processing on port 2)
>> SLB port 2# ../port 3/client ena (Enable client processing on port 3)
>> SLB port 3# apply (Apply the configuration)
>> SLB port 3# save (Save the configuration)
```

8. Configure the virtual server that will load balance the real servers.

```
>> SLB port 3# ../virt 100 (Configure virtual server 100)
>> Virtual Server 100# vip 20.1.1.10 (Assign virtual server 100 IP address)
>> Virtual Server 100# ena (Enable the virtual server)
```

9. Configure the real servers to which traffic will be load-balanced.

These are the real servers on the network.

```
>> Real server group 1# /cfg/slb/real/2 (Select real server 2)
>> Real server 2 # rip 20.1.1.2 (Assign real server 2 IP address)
>> Real server 2 # ena (Enable real server 2)
>> Real server 2 # ../real 3 (Select real server 3)
>> Real server 3 # rip 20.1.1.3 (Assign real server 3 IP address)
```

10. Place the real servers into a real server group.

```
>> Real server 3# ../group 200 (Select real server group 1)
>> Real server group 200# add 2 (Select real server 2 to group 200)
>> Real server group 200# add 3 (Select real server 3 to group 200)
```

11. Configure ports 4 and 5, which are connected to the real servers, for server processing.

```
>> Real server group 200# ../port 4/server ena
>> SLB port 4# ../port 5/server ena
```

12. Enable server load balancing on the switch.

```
>> Real server group 1# /cfg/slb/on
```

13. Create a filter to prevent server-to-server traffic from being redirected.

>> Layer 4# /cfg/slb/filt 10	(Select filter 10)
>> Filter 10# sip any	(From any source IP address)
>> Filter 10# dip 20.1.1.0	(To base IP address for IF 5)
>> Filter 10# dmask 255.255.255.0	(For the range of addresses)
>> Filter 10# proto any	(For any protocol)
>> Filter 10# action allow	(Allow traffic)
>> Filter 10# ena	(Enable the filter)

14. Create the redirection filter.

This filter will redirect outbound traffic, load balancing it among the defined real servers in the group. In this case, the real servers represent IP interfaces on the dirty-side switch.

>> Filter 10# ../filt 15	(Select filter 15)
>> Filter 15# sip any	(From any source IP address)
>> Filter 15# dip any	(To any destination IP address)
>> Filter 15# proto any	(For any protocol)
>> Filter 15# action redir	(Perform redirection)
>> Filter 15# group 1	(To real server group 1)
>> Filter 15# ena	(Enable the filter)

15. Add the filters to the ingress ports for the outbound packets.

Redirection filters are needed on all the ingress ports on the clean-side application switch. Ingress ports are any that attach to real servers or internal clients on the clean-side of the network. In this case, two real servers are attached to the clean-side application switch on port 4 and port 5.

>> Filter 15# ../port 4	(Select ingress port 4)
>> SLB Port 4# add 10	(Add the filter to the ingress port)
>> SLB Port 4# add 15	(Add the filter to the ingress port)
>> SLB Port 4# filt ena	(Enable filtering on the port)
>> SLB Port 4# ../port 5	(Select ingress port 5)
>> SLB Port 5# add 10	(Add the filter to the ingress port)
>> SLB Port 5# add 15	(Add the filter to the ingress port)
>> SLB Port 5# filt ena	(Enable filtering on the port)

16. Define static routes to the dirty-side IP interfaces, using the firewalls as gateways.

One static route is required for each firewall path being load balanced. In this case, two paths are required: one that leads to dirty-side IF 2 (10.1.1.1) through the first firewall (10.1.3.10) as its gateway, and one that leads to dirty-side IF 3 (10.1.2.1) through the second firewall (10.1.4.10) as its gateway.

```
>> SLB Port 5# /cfg/ip/route
>> IP Static Route# add 10.1.1.1 255.255.255.255 10.1.3.10
>> IP Static Route# add 10.1.2.1 255.255.255.255 10.1.4.10
```

NOTE – Configuring static routes for FWLB does not require IP forwarding to be turned on.

17. Apply and save the configuration changes.

```
>> # apply
>> # save
```

Four-Subnet FWLB

The four-subnet FWLB method is often deployed in large networks that require high-availability solutions. This method uses filtering, static routing, and Virtual Router Redundancy Protocol (VRRP) to provide parallel firewall operation between redundant application switches.

Figure 15-5 shows one possible network topology using the four-subnet method:

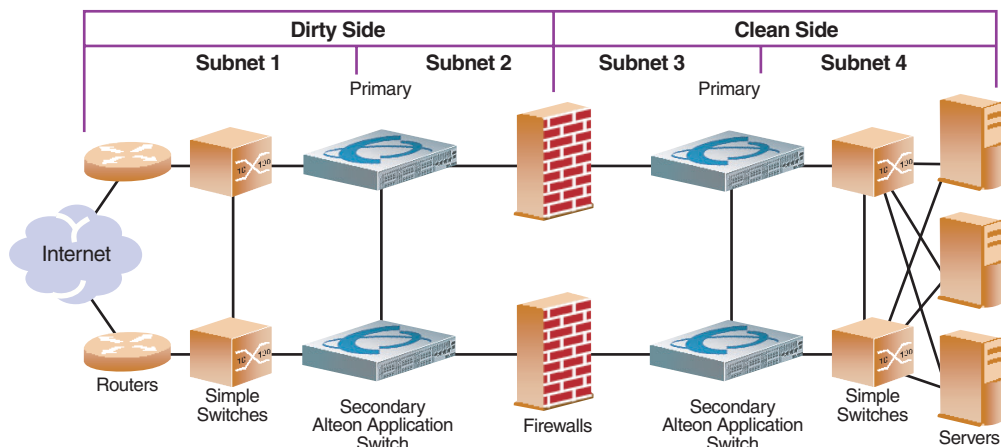


Figure 15-5 Four-Subnet FWLB Topology

This network is classified as a high-availability network because no single component or link failure could cause network resources to become unavailable. Simple switches and vertical block interswitch connections are used to provide multiple paths for network failover. However, the interswitch links may be trunked together with multiple ports for additional protection from failure.

NOTE – Other topologies that use internal hubs, or diagonal cross-connections between the Alteon Application Switches and simple switches are also possible. While such topologies may resolve networking issues in special circumstances, they can make configuration more complex and can cause restrictions on the use of advanced features such as Active-Active VRRP, free-metric FWLB, or Content Intelligent Switching. Alternate topologies are explored in more detail in Alteon OS FWLB white papers, but are not within the scope of this manual.

As shown in [Figure 15-5](#), the network is divided into four sections:

- Subnet 1 includes all equipment between the exterior routers and dirty-side application switches.
- Subnet 2 includes the dirty-side application switches with their interswitch link, and dirty-side firewall interfaces.
- Subnet 3 includes the clean-side firewall interfaces, and clean-side application switches with their interswitch link.
- Subnet 4 includes all equipment between the clean-side application switches and their servers.

In this network, external traffic arrives through both routers. Since VRRP is enabled, one of the dirty-side application switches acts as primary and receives all traffic. The dirty-side primary application switch performs FWLB in a fashion similar to basic FWLB: a redirection filter splits traffic into multiple streams which are routed through the available firewalls to the primary clean-side application switch.

Just as with the basic method, four-subnet FWLB uses the hash metric to distribute firewall traffic and maintain persistence, though other load-balancing metrics can be used by configuring an additional Return to Sender (RTS) option (see [“Free-Metric FWLB” on page 389](#)).

Four-Subnet FWLB Implementation

In this example, traffic between the redundant Alteon Application Switches is load balanced among the available firewalls.

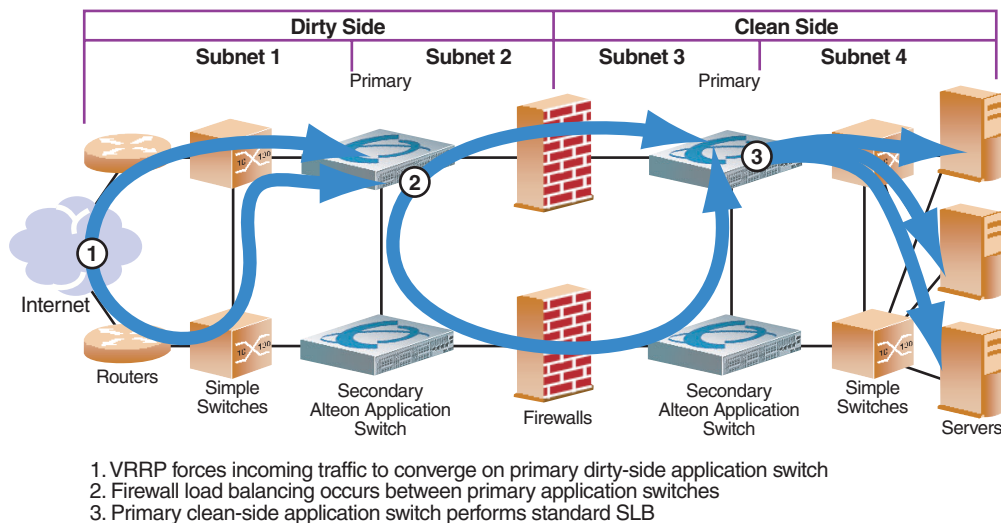


Figure 15-6 Four-Subnet FWLB Process

1. Incoming traffic converges on the primary dirty-side application switch.

External traffic arrives through redundant routers. A set of interconnected switches ensures that both routers have a path to each dirty-side application switch.

VRRP is configured on each dirty-side application switch so that one acts as the primary routing switch. If the primary fails, the secondary takes over.

2. FWLB is performed between primary application switches.

Just as with basic FWLB, filters on the ingress ports of the dirty-side application switch redirect traffic to a real server group composed of multiple IP addresses. This configuration splits incoming traffic into multiple streams. Each stream is then routed toward the primary clean-side application switch through a different firewall.

Although other load balancing metrics can be used in some configurations (see [“Free-Metric FWLB” on page 389](#)), the distribution of traffic within each stream is normally based on a mathematical *hash* of the IP source and destination addresses. Hashing ensures that each request and its related responses will use the same firewall (a feature known as *persistence*), and that the streams will be statistically equal in traffic load.

3. The primary clean-side application switch forwards the traffic to its destination.

Once traffic arrives at the primary clean-side application switch, it is forwarded to its destination. In this example, the application switch uses regular server load balancing settings to select a real server on the internal network for each incoming request.

The same process is used for outbound server responses; a filter on the clean-side application switch splits the traffic, and static routes forward each response stream back through the same firewall that forwarded the original request.

Configuring Four-Subnet FWLB

An example network for four-subnet FWLB is illustrated in Figure 15-7. While other complex topologies are possible, this example assumes a high-availability network using block (rather than diagonal) interconnections between switches.

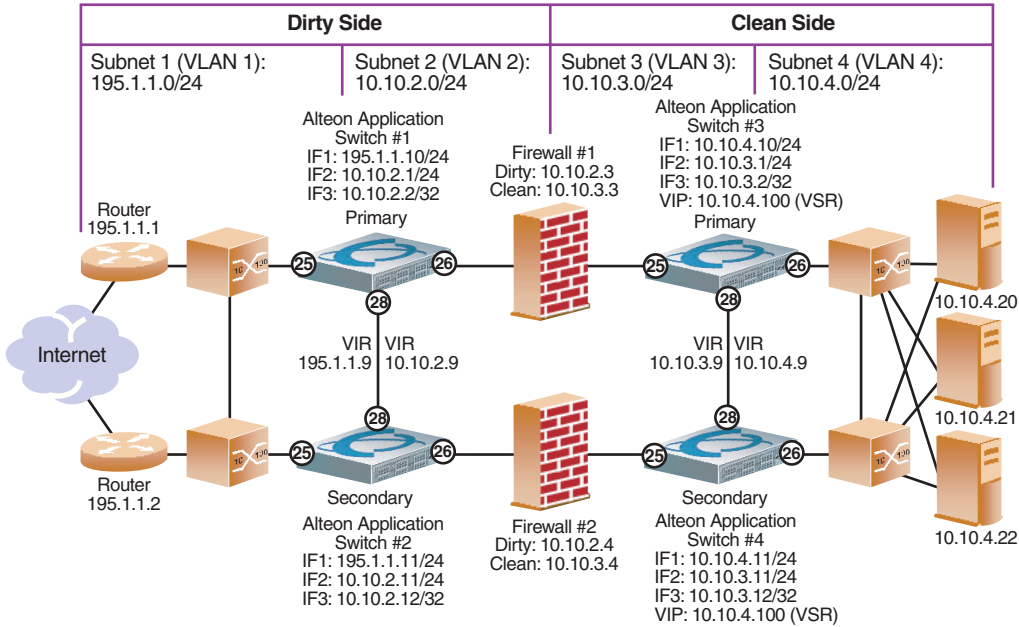


Figure 15-7 Four-Subnet FWLB Example Network

NOTE – The port designations of both dirty-side application switches are identical, as are the port designations of both clean-side application switches. This simplifies configuration by allowing you to synchronize each primary application switch’s configuration with the secondary.

Four-subnet FWLB configuration is summarized as follows:

- Configure routers and firewalls and test them for proper operation.
- Configure VLANs, IP interfaces, and static routes on all application switches and test them.
- Configure secondary application switches with VRRP support settings.
- Configure FWLB groups and redirection filters on the primary dirty-side application switch.
- Configure and synchronize VRRP on the primary dirty-side application switch.
- Configure FWLB and SLB groups, and add FWLB redirection filters on the primary clean-side application switch.
- Configure VRRP on the primary clean-side application switch and synchronize the secondary.

These steps are explained in detail in the following sections.

Configure the Routers

The routers must be configured with a static route to the destination services being accessed by the external clients.

In this example, the external clients intend to connect to services at a publicly advertised IP address on this network. Since the real servers are load balanced behind a virtual server on the clean-side application switch using normal SLB settings, the routers require a static route to the virtual server IP address. The next hop for this static route is the application switch Virtual Interface Router (VIR), which is in the same subnet as the routers:

```
Route Added: 10.10.4.100 (to clean-side virtual server) via
195.1.1.9 (Subnet 1 VIR)
```

Configure the Firewalls

Before you configure the Alteon Application Switches, the firewalls must be properly configured. For incoming traffic, each firewall must be configured with a static route to the clean-side virtual server, using the VIR in its clean-side subnet as the next hop. For outbound traffic, each firewall must use the VIR in its dirty-side subnet as the default gateway.

In this example, the firewalls are configured with the following IP addresses:

Table 2 Four-Subnet Firewall IP Address Configuration

Item	Address
Firewall 1	
Dirty-side IP interface	10.10.2.3
Clean-side IP interface	10.10.3.3
Default Gateway	10.10.2.9 (Subnet 2 VIR)
Route Added	10.10.4.100 (virtual server) via 10.10.3.9 (Subnet 3 VIR)
Firewall 2	
Dirty-side IP interface	10.10.2.4
Clean-side IP interface	10.10.3.4
Default Gateway	10.10.2.9 (dirty-side VIR)
Route Added	10.10.4.100 (virtual server) via 10.10.3.9 (Subnet 3 VIR)

The firewalls must also be configured with rules that determine which types of traffic will be forwarded through the firewall and which will be dropped. All firewalls participating in FWLB must be configured with the same set of rules.

NOTE – It is important to test the behavior of the firewalls prior to adding FWLB.

Configure Connectivity for the Primary Dirty-Side Switch

1. Configure VLANs on the primary dirty-side application switch.

Two VLANs are required. VLAN 1 includes port 25, for the Internet connection. VLAN 2 includes port 26, for the firewall connection, and port 28, for the interswitch connection.

```
>> # /cfg/vlan 2
>> # add 26
>> # add 28
>> # ena
```

(Port 26 connects to the firewall)
(Port 28 is the inter-switch connection)

NOTE – Port 25 is part of VLAN 1 by default and does not require manual configuration.

2. Configure IP interfaces on the primary dirty-side application switch.

Three IP interfaces (IF's) are used. IF 1 is on placed on Subnet 1. IF 2 will be used for routing traffic through the top firewall. IF 3 will be used for routing traffic through the lower firewall. To avoid confusion, IF 2 and IF 3 will be used in the same way on all application switches.

```
>> # /cfg/ip/if 1
>> # mask 255.255.255.0
>> # addr 195.1.1.10
>> # ena
>> # ../if 2
>> # mask 255.255.255.0
>> # addr 10.10.2.1
>> # vlan 2
>> # ena
>> # ../if 3
>> # mask 255.255.255.255
>> # addr 10.10.2.2
>> # vlan 2
>> # ena
```

NOTE – By configuring the IP interface mask prior to the IP address, the broadcast address is automatically calculated. Also, only the first IP interface in a given subnet is given the full subnet range mask. Subsequent IP interfaces (such as IF 3) are given individual masks.

3. Turn Spanning Tree Protocol (STP) off for the primary dirty-side application switch.

```
>> # /cfg/stp #/off
```

4. Configure static routes on the primary dirty-side application switch.

Four static routes are required:

- To primary clean-side IF 2 via Firewall 1 using dirty-side IF 2
- To primary clean-side IF 3 via Firewall 2 using dirty-side IF 3
- To secondary clean-side IF 2 via Firewall 1 using dirty-side IF 2
- To secondary clean-side IF 3 via Firewall 2 using dirty-side IF 3

NOTE – Remember, IF 2 is being used on all application switches whenever routing through the top firewall, and IF 3 is being used on all application switches whenever routing through the lower firewall.

The static route add command uses the following format:

add <destination address> <dest. mask> <gateway address> <source interface>

This example requires the following static route configuration:

```
>> # /cfg/ip/frwd/route
>> # add 10.10.3.1 255.255.255.255 10.10.2.3 2
>> # add 10.10.3.2 255.255.255.255 10.10.2.4 3
>> # add 10.10.3.11 255.255.255.255 10.10.2.3 2
>> # add 10.10.3.12 255.255.255.255 10.10.2.4 3
```

NOTE – When defining static routes for FWLB, it is important to specify the source IP interface numbers.

5. When dynamic routing protocols are not used, configure a gateway to the external routers.

```
>> # cfg/ip/gw 1/addr 195.1.1.1
>> # cfg/ip/gw 2/addr 195.1.1.2
```

6. Make your changes take effect.

```
>> # apply
>> # save
>> # /boot/reset
```

Configure Connectivity for the Secondary Dirty-Side Switch

Except for the IP interfaces, this configuration is identical to the primary dirty-side application switch.

1. Configure VLANs on the secondary dirty-side application switch.

```
>> # /cfg/vlan 2
>> # add 26
>> # add 28
>> # ena
```

2. Configure IP interfaces on the secondary dirty-side application switch.

```
>> # /cfg/ip/if 1
>> # mask 255.255.255.0
>> # addr 195.1.1.11
>> # ena
>> # ../if 2
>> # mask 255.255.255.0
>> # addr 10.10.2.11
>> # vlan 2
>> # ena
>> # ../if 3
>> # mask 255.255.255.255
>> # addr 10.10.2.12
>> # vlan 2
>> # ena
```

3. Turn STP off for the secondary dirty-side application switch.

```
>> # /cfg/stp #/off
```

4. Configure static routes on the secondary dirty-side application switch.

```
>> # /cfg/ip/frwd/route
>> # add 10.10.3.1 255.255.255.255 10.10.2.3 2
>> # add 10.10.3.2 255.255.255.255 10.10.2.4 3
>> # add 10.10.3.11 255.255.255.255 10.10.2.3 2
>> # add 10.10.3.12 255.255.255.255 10.10.2.4 3
```


5. When dynamic routing protocols are not used, configure a gateway to the external routers on the secondary dirty-side switch.

```
>> # cfg/ip/gw 1/addr 195.1.1.1
>> # cfg/ip/gw 2/addr 195.1.1.2
```

6. Apply and save your configuration.

```
>> # apply
>> # save
>> # /boot/reset
```

Configure Connectivity for the Primary Clean-Side Switch

1. Configure VLANs on the primary clean-side application switch.

Two VLANs are required. VLAN 3 includes the firewall port and interswitch connection port. VLAN 4 includes the port that attaches to the real servers.

```
>> # /cfg/vlan 3
>> # add 25
>> # add 28
>> # ena
>> # ../vlan 4
>> # add 26
>> # ena
```

2. Configure IP interfaces on the primary clean-side application switch.

```
>> # /cfg/ip/if 1
>> # mask 255.255.255.0
>> # addr 10.10.4.10
>> # vlan 4
>> # ena
>> # ../if 2
>> # mask 255.255.255.0
>> # addr 10.10.3.1
>> # vlan 3
>> # ena
>> # ../if 3
>> # mask 255.255.255.255
>> # addr 10.10.3.2
>> # vlan 3
>> # ena
```

3. Turn STP off for the primary clean-side application switch.

```
>> # /cfg/stp/off
```

Spanning Tree Protocol is disabled because VLANs prevent broadcast loops.

4. Configure static routes on the primary clean-side application switch.

Four static routes are needed:

- To primary dirty-side IF 2 via Firewall 1 using clean-side IF 2
- To primary dirty-side IF 3 via Firewall 2 using clean-side IF 3
- To secondary dirty-side IF 2 via Firewall 1 using clean-side IF 2
- To secondary dirty-side IF 3 via Firewall 2 using clean-side IF 3

The static route add command uses the following format:

```
add <destination address> <dest. mask> <gateway address> <source interface>
```

This example requires the following static route configuration:

```
>> # /cfg/ip/frwd/route
>> # add 10.10.2.1 255.255.255.255 10.10.3.3 2
>> # add 10.10.2.2 255.255.255.255 10.10.3.4 3
>> # add 10.10.2.11 255.255.255.255 10.10.3.3 2
>> # add 10.10.2.12 255.255.255.255 10.10.3.4 3
```

5. Apply and save your changes.

```
>> # apply
>> # save
>> # /boot/reset
```

Configure Connectivity for the Secondary Clean-Side Switch

1. Configure VLANs on the secondary clean-side application switch.

```
>> # /cfg/vlan 3
>> # add 25
>> # add 28
>> # ena
>> # ../vlan 4
>> # add 26
>> # ena
```

2. Configure IP interfaces on the secondary clean-side application switch.

```
>> # /cfg/ip/if 1
>> # mask 255.255.255.0
>> # addr 10.10.4.11
>> # vlan 4
>> # ena
>> # ../if 2
>> # mask 255.255.255.0
>> # addr 10.10.3.11
>> # vlan 3
>> # ena
>> # ../if 3
>> # mask 255.255.255.255
>> # addr 10.10.3.12
>> # vlan 3
>> # ena
```

3. Turn STP off for the secondary clean-side application switch.

```
>> # /cfg/stp/off
```

Spanning Tree Protocol is disabled because VLANs prevent broadcast loops.

4. Configure static routes on the secondary clean-side application switch.

```
>> # /cfg/ip/frwd/route
>> # add 10.10.2.1 255.255.255.255 10.10.3.3 2
>> # add 10.10.2.2 255.255.255.255 10.10.3.4 3
>> # add 10.10.2.11 255.255.255.255 10.10.3.3 2
>> # add 10.10.2.12 255.255.255.255 10.10.3.4 3
```

5. Apply and save your changes.

```
>> # apply
>> # save
>> # /boot/reset
```

Verify Proper Connectivity

To verify proper configuration up to this point, use the `ping` option to test network connectivity. At each Alteon Application Switch, you should receive a valid response when pinging the destination addresses established in the static routes.

For example, on the secondary clean-side application switch, the following commands should receive a valid response:

```
>> # ping 10.10.2.1
Response; 10.10.2.1: #1 OK, RTT 1 msec.
>> # ping 10.10.2.2
Response; 10.10.2.2: #1 OK, RTT 1 msec.
>> # ping 10.10.2.11
Response; 10.10.2.11: #1 OK, RTT 1 msec.
>> # ping 10.10.2.12
Response; 10.10.2.12: #1 OK, RTT 1 msec.
```

Configure VRRP Support on the Secondary Dirty-Side Switch

The secondary dirty-side application switch must be configured with the primary as its peer. Once this is done, the secondary application switch will get the remainder of its configuration from the primary when synchronized in a later step.

In this example, the secondary application switch is configured to use primary dirty-side interface 1 as its peer.

```
>> # /cfg/vrrp/on
>> # /cfg/slb
>> # on
>> # sync/peer 1
>> # addr 195.1.1.10
>> # ena
>> # apply
>> # save
```

Configure VRRP Support on the Secondary Clean-Side Switch

In this example, the secondary application switch uses primary clean-side interface 1 as its peer.

```
>> # /cfg/vrrp/on
>> # /cfg/slb
>> # on
>> # sync/peer 1
>> # addr 10.10.4.10
>> # ena
>> # apply
>> # save
```

Complete the Configuration of the Primary Dirty-Side Switch

1. Create an FWLB real server group on the primary dirty-side application switch.

A real server group is used as the target for the FWLB redirection filter. Each IP address that is assigned to the group represents a path through a different firewall. In this case, since two firewalls are used, two addresses are added to the group.

Earlier, it was stated that this example uses IF 2 on all application switches whenever routing through the top firewall, and IF 3 on all application switches whenever routing through the lower firewall. Therefore, the first address will represent the primary clean-side IF 2, and the second represents the primary clean-side IF 3.

```
>> # /cfg/slb
>> # on
>> # real 1
>> # rip 10.10.3.1
>> # ena
>> # ../real 2
>> # rip 10.10.3.2
>> # ena
>> # ../group 1
>> # add 1
>> # add 2
>> # metric hash
```

Using the hash metric, all traffic between specific IP source/destination address pairs flows through the same firewall, ensuring that sessions established by the firewalls are maintained for their duration (persistence).

NOTE – Other load balancing metrics, such as `leastconns`, `roundrobin`, `minmiss`, `response`, and `bandwidth`, can be used when enabling the Return to Sender (RTS) option. For more information, see “Free-Metric FWLB” on page 389.

2. Create the FWLB filters.

Three filters are required on the port attaching to the routers:

- Filter 10 prevents local traffic from being redirected.
- Filter 20 prevents VRRP traffic (and other multicast traffic on the reserved 224.0.0.0/24 network) from being redirected.
- Filter 2048 redirects the remaining traffic to the firewall group.

```
>> # /cfg/slb/filt 10
>> # dip 195.1.1.0
>> # dmask 255.255.255.0
>> # ena
>> # ../filt 20
>> # dip 224.0.0.0
>> # dmask 255.255.255.0
>> # ena
>> # ../filt 2048
>> # action redir
>> # group 1
>> # ena
>> # ../port 1
>> # filt ena
>> # add 10
>> # add 20
>> # add 2048
```

3. Configure VRRP on the primary dirty-side application switch.

VRRP in this example requires two virtual routers—one for the subnet attached to the routers, and one for the subnet attached to the firewalls.

```
>> # /cfg/vrrp
>> # on
>> # vr 1
>> # vrid 1                                (Configure virtual router 1)
>> # addr 195.1.1.9                        (For the subnet attached to the routers)
>> # if 1
>> # prio 101
>> # share dis
>> # ena
>> # track
>> # ifs ena
>> # ports ena
>> # /cfg/vrrp/vr 2
>> # vrid 2                                (Configure virtual router 2)
>> # addr 10.10.2.9                        (For the subnet attached to the firewall)
>> # if 2
>> # prio 101
>> # share dis
>> # ena
>> # track
>> # ifs ena
>> # ports ena
```

4. Configure the VRRP peer on the primary dirty-side application switch.

```
>> # /cfg/slb/sync
>> # prios d
>> # peer 1
>> # ena
>> # addr 195.1.1.11
```

5. Apply and save your configuration changes.

```
>> # apply
>> # save
```

6. Synchronize primary and secondary dirty-side application switches for the VRRP configuration.

```
>> # /oper/slb/sync
```

Complete the Configuration of the Primary Clean-Side Switch

1. Create an FWLB real server group on the primary clean-side application switch.

A real server group is used as the target for the FWLB redirection filter. Each IP address assigned to the group represents a return path through a different firewall. In this case, since two firewalls are used, two addresses are added to the group. The two addresses are the interfaces of the dirty-side application switch, and are configured as if they are real servers.

NOTE – IF 2 is used on all application switches whenever routing through the top firewall, and IF 3 is used on all application switches whenever routing through the lower firewall.

```
>> # /cfg/slb
>> # on
>> # real 1
>> # rip 10.10.2.1                (IF2 of the primary dirty-side application switch)
>> # ena
>> # ../real 2
>> # rip 10.10.2.2                (IF2 of the primary dirty-side application switch)
>> # ena
>> # ../group 1
>> # add 1
>> # add 2
>> # metric hash
```

NOTE – The clean-side application switch must use the same metric as defined on the dirty side. For information on using metrics other than hash, see “Free-Metric FWLB” on page 389.

2. Create an SLB real server group on the primary clean-side application switch, to which traffic will be load-balanced.

The external clients intend to connect to HTTP services at a publicly advertised IP address. The servers on this network are load balanced by a virtual server on the clean-side application switch. SLB options are configured as follows:

>> # /cfg/slb	(Select the SLB menu)
>> # real 20	(Select real server 20)
>> # rip 10.10.4.20	(Set IP address of real server 20)
>> # ena	(Enable)
>> # ../real 21	(Select real server 21)
>> # rip 10.10.4.21	(Set IP address of real server 21)
>> # ena	(Enable)
>> # ../real 22	(Select real server 22)
>> # rip 10.10.4.22	(Set IP address of real server 22)
>> # ena	(Enable)
>> # ../group 2	(Select real server group 2)
>> # add 20	(Add the real servers to the group)
>> # add 21	
>> # add 22	
>> # metric leastconns	(Select least connections as the load balancing metric)
>> # ../virt 1	(Select the virtual server 1 menu)
>> # vip 10.10.4.100	(Set the virtual server IP address)
>> # service http	(Select HTTP for load balancing)
>> # group 2	(Add real server group 2)
>> # ena	(Enable the virtual server)
>> # ../port 26/server ena	(Enable server processing on the port connected to the real servers)
>> # ../port 25/client ena	(Enable client processing on the port connected to the firewall)
>> # ../port 28/client ena	(Enable client processing on the inter-switch connection)

NOTE – The virtual server IP address configured in this step will also be configured as a Virtual Server Router (VSR) when VRRP is configured in a later step.

3. Create the FWLB filters on the primary clean-side application switch.

Three filters are required on the port attaching to the real servers:

- Filter 10 prevents local traffic from being redirected.
- Filter 20 prevents VRRP traffic from being redirected.
- Filter 2048 redirects the remaining traffic to the firewall group.

```
>> # /cfg/slb/filt 10
>> # dip 10.10.4.0
>> # dmask 255.255.255.0
>> # ena
>> # ../filt 20
>> # dip 224.0.0.0
>> # dmask 255.255.255.0
>> # ena
>> # ../filt 2048
>> # action redir
>> # group 1
>> # ena
>> # ../port 4
>> # filt ena
>> # add 10
>> # add 20
>> # add 2048
```

4. Configure VRRP on the primary clean-side application switch.

VRRP in this example requires two virtual routers to be configured—one for the subnet attached to the real servers, and one for the subnet attached to the firewalls.

```
>> # /cfg/vrrp
>> # on
>> # vr 1
>> # vrid 3
>> # addr 10.10.4.9
>> # if 1
>> # prio 100
>> # share dis
>> # ena
>> # track
>> # ifs ena
>> # ports ena
>> # /cfg/vrrp/vr 2
>> # vrid 4
>> # addr 10.10.3.9
>> # if 2
>> # prio 101
>> # share dis
>> # ena
>> # track
>> # ifs ena
>> # ports ena
```

A third virtual router is required for the virtual server used for optional SLB.

```
>> # /cfg/vrrp/vr 3
>> # vrid 5
>> # addr 10.10.4.100
>> # prio 102
>> # share dis
>> # ena
>> # track
>> # ifs ena
>> # ports ena
```

5. Configure the peer on the primary clean-side application switch.

```
>> # /cfg/slb/sync  
>> # prios d  
>> # peer 1  
>> # ena  
>> # addr 10.10.4.11
```

6. Apply and save your configuration changes.

```
>> # apply  
>> # save
```

7. Synchronize primary and secondary dirty-side application switches for the VRRP configuration.

```
>> # /oper/slb/sync
```

Advanced FWLB Concepts

Free-Metric FWLB

Free-metric FWLB allows you to use load-balancing metrics other than hash, such as `leastconns`, `roundrobin`, `minmiss`, `response`, and `bandwidth` for more versatility. The free-metric method uses the Return to Sender (RTS) option. RTS can be used with basic FWLB or four-subnet FWLB networks.

Free-Metric with Basic FWLB

For this example, review the basic FWLB example network.

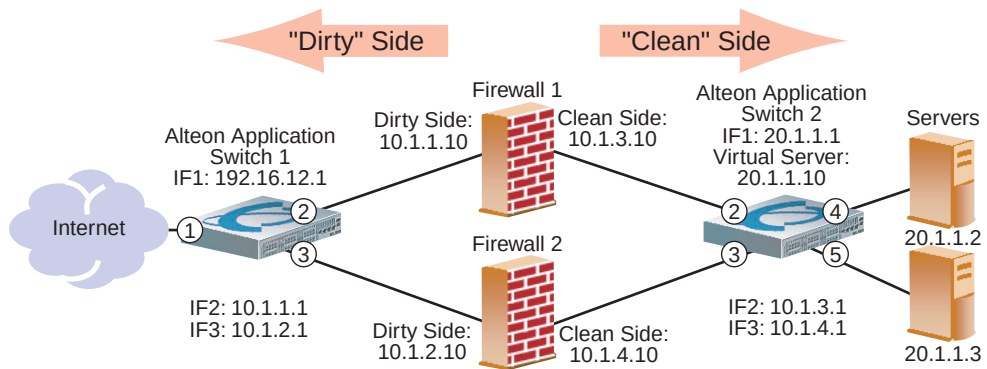


Figure 15-8 Basic FWLB Example Network

To use free-metric FWLB in this network, the following configuration changes are necessary.

1. **On the clean-side application switch, enable RTS on the ports attached to the firewalls (ports 2 and 3).**

Enable filter and server processing on ports 2 and 3, so that the responses from the real server are looked up in the session table.

```
>> # /cfg/slb/port 2/rts enable
>> # ../port 3/rts enable
```

2. **On the clean-side application switch, remove the redirection filter from the ports attached to the real servers (ports 4 and 5), but make sure filter processing is enabled.**

The redirection filter is removed so that the return packet traverses through the same firewall. If the firewalls synchronize their states, then it is not required to remove the redirection filter. Filter processing is enabled to make use of the RTS-created sessions.

```
>> # ../port 4/rem 2048
>> # filt ena
>> # ../port 5/rem 2048
>> # filt ena
```

Make sure to use the hash metric if the session is from an FTP or RTSP servers.

3. **On the dirty-side application switch, set the FWLB metric.**

```
>> # ../group 1
>> # metric <metric type>
```

Any of the following load-balancing metrics can be used: hash, leastconns, roundrobin, minmiss, response, and bandwidth. See [“Metrics for Real Server Groups” on page 151](#) for details on using each metric.

NOTE – Some metrics allow other options (such as weights) to be configured.

Free-Metric with Four-Subnet FWLB

For this example, review the four-subnet example network.

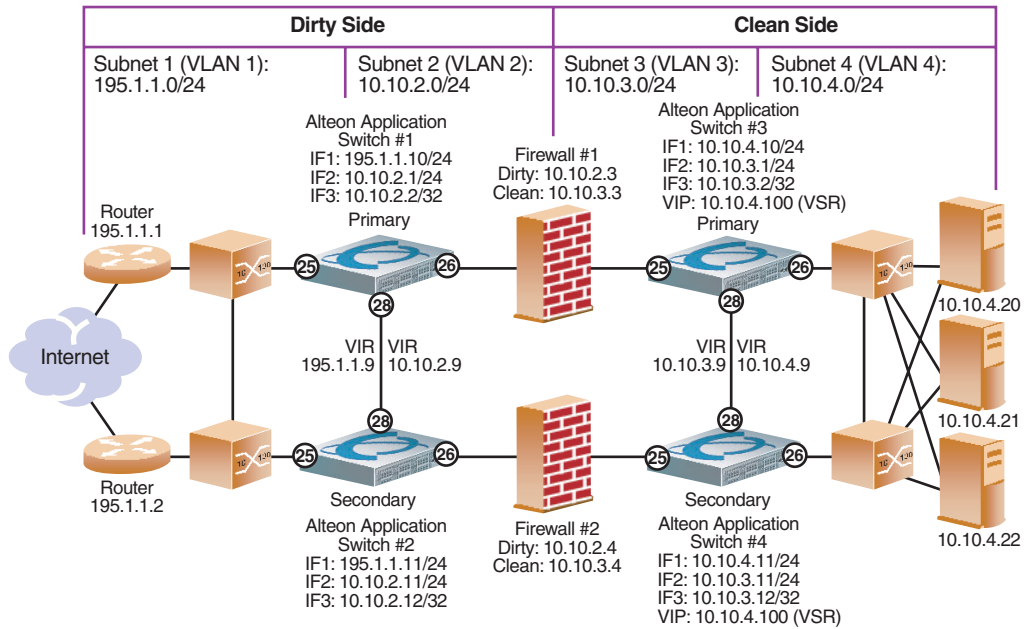


Figure 15-9 Four-Subnet FWLB Example Network

To use free-metric FWLB in this network, the following configuration changes are necessary.

1. **On the clean-side application switches, enable RTS on the ports attached to the firewalls (port 3) and on the interswitch port (port 9).**

Enable filter and server processing on ports 3 and 9, so that the responses from the real server are looked up in the session table.

On *both* clean-side switches:

```
>> # /cfg/slb/port 3/rts enable
>> # ../port 9/rts enable
```

2. **On the clean-side application switches, remove the redirection filter from the ports attached to the real servers (ports 4), but make sure filter processing is enabled.**

To view the original redirection filters that were configured for the four-subnet example, see [Step 3. on page 386](#).

On *both* clean-side switches:

```
>> # ../port 4/rem 2048
>> # filt ena
```

3. **On the dirty-side application switches, set the FWLB metric.**

On *both* dirty-side application switches:

```
>> # ../group 1
>> # metric <metric type>
```

Any of the following load-balancing metrics can be used: hash, leastconns, roundrobin, minmiss, response, and bandwidth. See [“Metrics for Real Server Groups” on page 151](#) for details on using each metric.

NOTE – Some metrics allow other options (such as weights) to be configured.

Adding a Demilitarized Zone (DMZ)

Implementing a DMZ in conjunction with firewall load balancing enables the Alteon Application Switch to do the traffic filtering, off-loading this task from the firewall. A DMZ is created by configuring FWLB with another real server group and a redirection filter towards the DMZ subnets.

The DMZ servers can be connected to the application switch on the dirty side of the firewall. A typical firewall load balancing configuration with a DMZ is shown in [Figure 15-10](#).

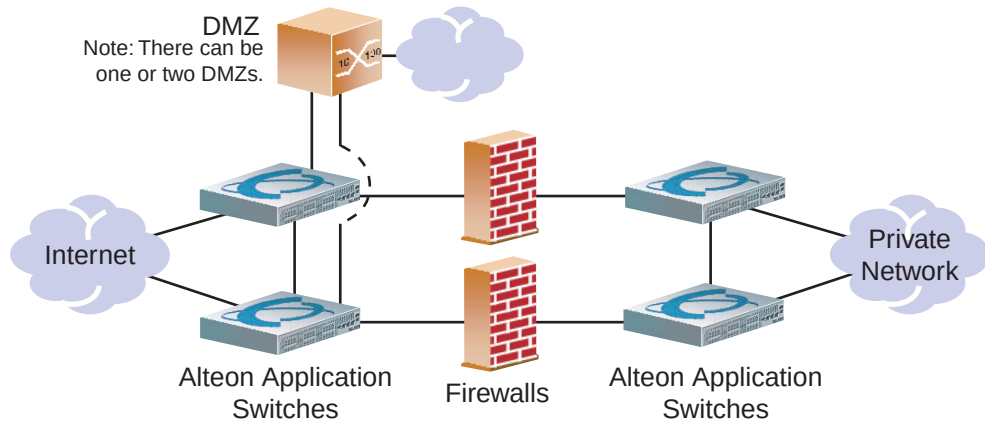


Figure 15-10 Typical Firewall Load-Balancing Topology with DMZ

The DMZ servers can be attached to the application switch directly or through an intermediate hub or switch. The application switch is then configured with filters to permit or deny access to the DMZ servers. In this manner, two levels of security are implemented: one that restricts access to the DMZ through the use of application switch filters, and another that restricts access to the clean network through the use of stateful inspection performed by the firewalls.

You could add the filters required for the DMZ (to each application switch) as follows:

- 1. On the dirty-side application switch, create the filter to allow HTTP traffic to reach the DMZ Web servers.**

In this example, the DMZ Web servers use IP addresses 205.178.29.0/24.

```
>> # /cfg/slb/filt 80                (Select filter 80)
>> Filter 80# sip any                (From any source IP address)
>> Filter 80# dip 205.178.29.0      (To the DMZ base destination)
>> Filter 80# dmask 255.255.255.0  (For the range of DMZ addresses)
>> Filter 80# proto tcp              (For TCP protocol traffic)
>> Filter 80# sport any              (From any source port)
>> Filter 80# dport http             (To an HTTP destination port)
>> Filter 80# action allow           (Allow the traffic)
>> Filter 80# ena                    (Enable the filter)
```

- 2. Create another filter to deny all other traffic to the DMZ Web servers.**

```
>> Filter 80# ../filt 89             (Select filter 89)
>> Filter 89# sip any                (From any source IP address)
>> Filter 89# dip 205.178.29.0      (To the DMZ base destination)
>> Filter 89# dmask 255.255.255.0  (For the range of DMZ addresses)
>> Filter 89# proto any              (For TCP protocol traffic)
>> Filter 89# action deny           (Allow the traffic)
>> Filter 89# ena                    (Enable the filter)
```

NOTE – The deny filter has a higher filter number than the allow filter. This is necessary so that the allow filter has the higher order of precedence.

- 3. Add the filters to the traffic ingress ports.**

```
>> Filter 89# ../port 1              (Select the ingress port)
>> SLB Port 1# add 80                (Add the allow filter)
>> SLB Port 1# add 89                (Add the deny filter)
```

- 4. Apply and save the configuration changes.**

```
>> SLB Port 1# apply
>> SLB Port 1# save
```

Firewall Health Checks

Basic FWLB health checking is automatic. No special configuration is necessary unless you wish to tune the health checking parameters. See [Chapter 12, “Health Checking”](#) for details.

Firewall Service Monitoring

To maintain high availability, application switches monitor firewall health status and send packets only to healthy firewalls. There are two methods of firewall service monitoring: ICMP and HTTP. Each application switch monitors the health of the firewalls on a regular basis by pinging the IP interfaces configured on its partner application switch on the other side of the firewall.

If an application switch IP interface fails to respond to a user-specified number of pings, it (and, by implication, the associated firewall), is placed in a *Server Failed* state. At this time, the partner application switch stops routing traffic to that IP interface and, instead, distributes it across the remaining healthy application switch IP interfaces and firewalls.

When an application switch IP interface is in the *Server Failed* state, its partner application switch continues to send pings to it at user-configurable intervals. After a specified number of successful pings, the IP interface (and its associated firewall) is brought back into service.

For example, to configure the switch to allow one-second intervals between health checks or pings, two failed health checks to remove the firewall, and four successful health checks to restore the firewall to the real server group, use the following command:

```
>> /cfg/slb/real <real server number>/inter 1/retry 2/restr 4
```

Physical Link Monitoring

Alteon Application switches also monitor physical link status of switch ports connected to firewalls. If the physical link to a firewall goes down, that firewall is placed immediately in the *Server Failed* state. When an Alteon Application Switch detects that a failed physical link to a firewall has been restored, it brings the firewall back into service.

Using HTTP Health Checks

For those firewalls that do not permit ICMP pings to pass through, application switches can be configured to perform HTTP health checks, as described below.

1. Set the health check type to HTTP instead of ICMP.

```
>> # /cfg/slb/group 1/health http
```

(Select HTTP health checks)

2. Enable HTTP access to the switch.

```
>> # /cfg/sys/http ena
```

(Enable HTTP)

3. Configure a “dummy” redirect filter as the last filter (after the *redirect all* filter) to force the HTTP health checks to activate, as shown below:

```
>> # /cfg/slb/filt 2048
```

(Select filter 2048)

```
>> Filter 2048# proto tcp
```

(For TCP protocol traffic)

```
>> Filter 2048# action redir
```

(Redirect the traffic)

```
>> Filter 2048# group 1
```

(Set real server group for redirection)

```
>> Filter 2048# rport http
```

(Set real server port for redirection)

```
>> Filter 2048# ena
```

(Enable the filter)

NOTE – Make sure that the number of each real filter is lower than the number of the “dummy” redirect filter.

4. Apply filter to the port directed to the firewall.

```
>> # /cfg/slb/port #/add 2048
```

(Add the dummy filter)

In addition to HTTP, Alteon OS allows you to configure up to 5 different TCP services to listen for health checks. For example, you can configure FTP and SMTP ports to perform health checks. Refer to [Table 9-3 on page 148](#) for a list of other well-known application ports.



CHAPTER 16

Virtual Private Network Load Balancing

The VPN (Virtual Private Network) load balancing feature in Alteon OS allows the switch to load balance simultaneously up to 255 VPN devices. The switch records from which VPN server a session was initiated and ensures that the traffic returns back to the same VPN server from which the session started.

The following topics are addressed in this chapter:

- [“Overview” on page 398](#)

This section describes a VPN network and how VPN load balancing works on an Alteon Application Switch.

- [“VPN Load-Balancing Configuration” on page 400](#)

This section provides step-by-step instructions to configure VPN load balancing on a four-subnet network with four application switches and two VPN devices.

Overview

A VPN is a connection that has the appearance and advantages of a dedicated link, but it occurs over a shared network. Using a technique called *tunneling*, data packets are transmitted across a routed network, such as the Internet, in a private tunnel that simulates a point-to-point connection. This approach enables network traffic from many sources to travel via separate tunnels across the infrastructure. It also enables traffic from many sources to be differentiated, so that it can be directed to specific destinations and receive specific levels of service.

VPNs provide security features of a firewall, network address translation, data encryption, and authentication and authorization. Since most of the data sent between VPN initiators and terminators is encrypted, network devices cannot use information inside the packet to make intelligent routing decisions.

How VPN Load Balancing Works

VPN load balancing requires that all ingress traffic passing through a particular VPN must traverse the same VPN as it egresses back to the client. Traffic ingressing from the Internet is usually addressed to the VPNs, with the real destination encrypted inside the datagram. Traffic egressing the VPNs into the intranet contains the real destination in the clear.

In many VPN/firewall configurations it may not be possible to use the hash algorithm on the source and destination address, because the address may be encrypted inside the datagram. Also, the source/destination IP address of the packet may change as the packet traverses from the dirty-side switches to clean-side switches and back.

To support VPN load balancing, the Alteon Application Switch records state on frames entering the switch to and from the VPNs. This session table ensures that the same VPN server handles all the traffic between an inside host and an outside client for a particular session.

NOTE – VPN load balancing is supported for connecting from remote sites to the network behind the VPN cluster IP address. Connection initiated from clients internal to the VPN gateways is not supported.

Basic frame flow, from the dirty side of the network to the clean side, is shown in [Figure 16-1](#). An external client is accessing an internal server. The VPN devices do not perform Network Address Translation (NAT).

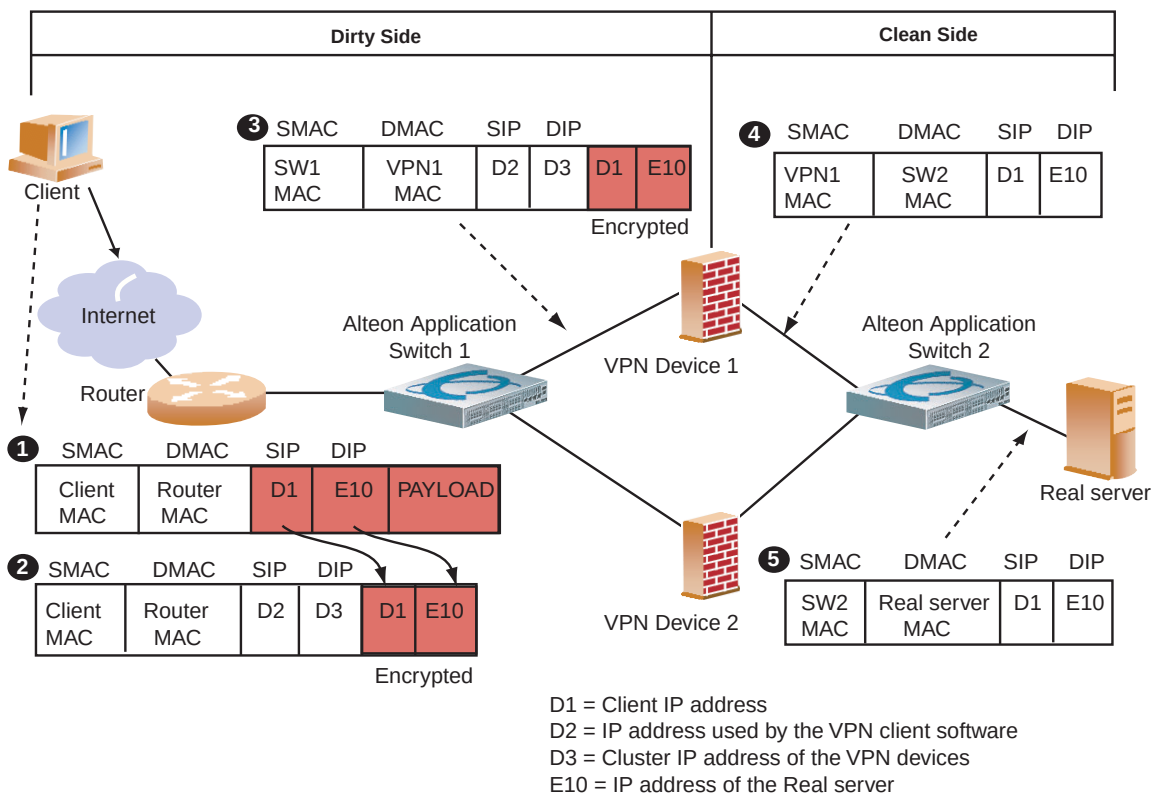


Figure 16-1 Basic Network Frame Flow and Operation

The basic steps that occur at the switches when a request arrives from the Internet are described below:

1. The client prepares to send traffic to the destination real server (with IP address E10).
2. The VPN client software encrypts the packet and sends it to the cluster IP address (D3) of the VPN devices.
3. Alteon Application Switch 1 makes an entry in the session table and forwards the packet to VPN device 1.
It is recommended to use the hash load-balancing metric to select the VPN device.
4. The VPN device 1 strips the IP header and decrypts the encrypted IP header.
5. Alteon Application Switch 2 forwards the packet to the real server.

If an entry is found, the frame is forwarded normally. If an entry is *not* found, the switch determines which VPN device processed the frame by performing a lookup with the source MAC address of the frame. If the MAC address matches a MAC address of a VPN device, the switch adds an entry to the session table so that reverse traffic is redirected to the same VPN device.

VPN Load-Balancing Configuration

Before you start configuring the switch for VPN load balancing, do the following:

- Configure the switch with firewall load balancing. For more information, see [“Firewall Load Balancing”](#) on page 353.
- Enable the Return to Sender (RTS) feature on the ports attached to the VPN devices, using the following command:

```
>> # /cfg/slb/port <port number>/rts ena
```

The following example illustrates VPN load balancing with two VPN devices and four Alteon Application Switches in a four subnet scenario.

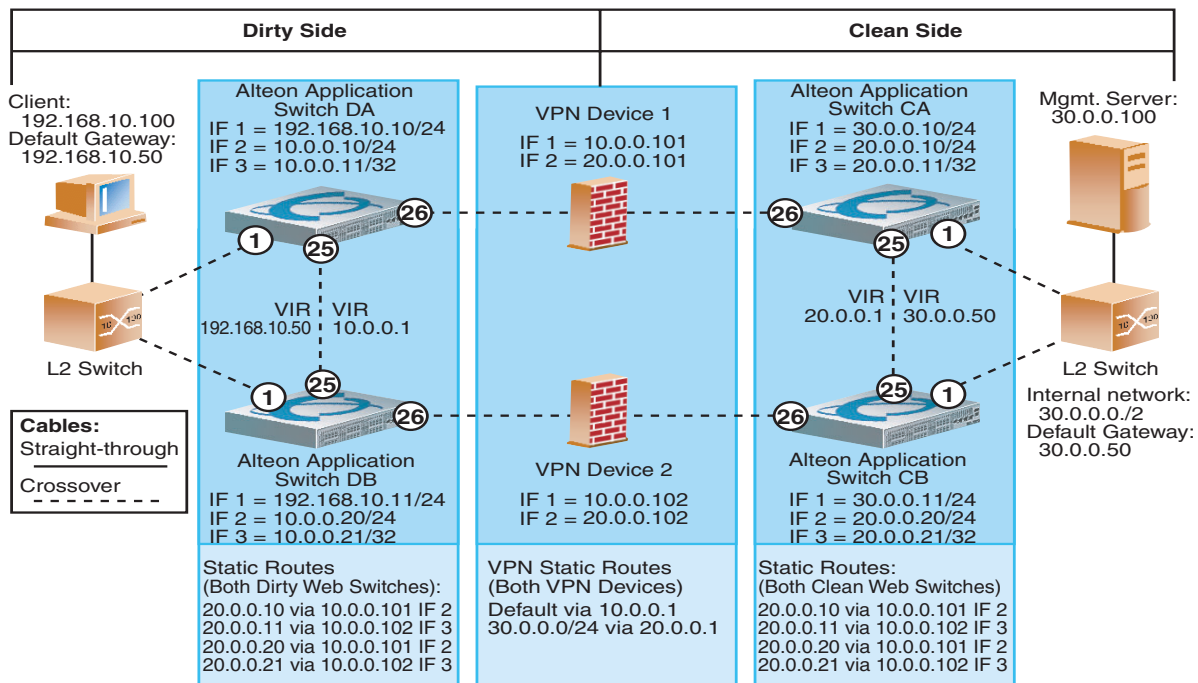


Figure 16-2 VPN Load-Balancing Configuration Example

Build the topology illustrated in [Figure 16-2 on page 400](#), and configure the switches as shown in the following sections.

Configure the First Clean-Side Application Switch-CA

1. Turn off BOOTP.

```
>> # /cfg/sys/bootp dis
```

2. Define and enable VLAN 2 for ports 25, and 26.

```
>> # /cfg/vlan 2/ena/def 25 26
```

3. Turn off Spanning Tree Protocol (STP).

```
>> # /cfg/stp/off
```

4. Define the clean-side IP interfaces.

Create one clean-side IP interface on a different subnet for each VPN device being load balanced.

>> # /cfg/ip/if 1/ena	<i>(Select IP interface 1 and enable)</i>
>> IP Interface 1# mask 255.255.255.0	<i>(Set subnet mask for interface 1)</i>
>> IP Interface 1# addr 30.0.0.10	<i>(Set IP address for interface 1)</i>
>> IP Interface 1# vlan 1	<i>(For VLAN 1)</i>
>> IP Interface 1# ../if 2/ena	<i>(Select IP interface 2 and enable)</i>
>> IP Interface 2# mask 255.255.255.0	<i>(Set subnet mask for interface 2)</i>
>> IP Interface 2# addr 20.0.0.10	<i>(Set IP address for interface 2)</i>
>> IP Interface 2# vlan 2	<i>(For VLAN 2)</i>
>> IP Interface 2# ../if 3/ena	<i>(Select IP interface 3 and enable)</i>
>> IP Interface 3# mask 255.255.255.255	<i>(Set subnet mask for interface 3)</i>
>> IP Interface 3# addr 20.0.0.11	<i>(Set IP address for interface 3)</i>
>> IP Interface 3# vlan 2	<i>(For VLAN 2)</i>

5. Configure routes for each of the IP interfaces you configured in Step 4 using the VPN devices as gateways.

One static route for redirection is required for each VPN device being load balanced.

```
>> # /cfg/ip/route
>> IP Static Route# add 10.0.0.10           (Static route destination IP address)
>> IP Static Route# 255.255.255.255        (Destination subnet mask)
>> IP Static Route# 20.0.0.101             (Enter gateway IP address)
>> IP Static Route# 2                      (For interface 2)
>> IP Static Route# add 10.0.0.11          (Enter destination IP address)
>> IP Static Route# 255.255.255.255        (Destination subnet mask)
>> IP Static Route# 20.0.0.102             (Enter gateway IP address)
>> IP Static Route# 3                      (For interface 3)
>> IP Static Route# add 10.0.0.20          (Enter destination IP address)
>> IP Static Route# 255.255.255.255        (Destination subnet mask)
>> IP Static Route# 20.0.0.101             (Enter gateway IP address)
>> IP Static Route# 2                      (For interface 2)
>> IP Static Route# add 10.0.0.21          (Static route destination IP address)
>> IP Static Route# 255.255.255.255        (Destination subnet mask)
>> IP Static Route# 20.0.0.102             (Enter gateway IP address)
>> IP Static Route# 3                      (For interface 3)
```

6. Configure VRRP for virtual routers 1 and 2.

```

>> # /cfg/vrrp/on                                     (Enable VRRP)
>> Virtual Router Redundancy Protocol# vr 1            (Select virtual router 1 menu)
>> VRRP Virtual Router 1# ena                         (Enable the virtual router)
>> VRRP Virtual Router 1# vrid 1                     (Assign virtual router ID 1)
>> VRRP Virtual Router 1# if 1                       (To interface number 1)
>> VRRP Virtual Router 1# prio 101                   (Set the renter priority)
>> VRRP Virtual Router 1# addr 30.0.0.50             (Set IP address of virtual router)
>> VRRP Virtual Router 1# share dis                  (Disable sharing)
>> VRRP Virtual Router 1# track                      (Select virtual router tracking menu)
>> VRRP VR 1 Priority Tracking# vrs ena               (Enable tracking of virtual routers)
>> VRRP VR 1 Priority Tracking# apply                 (Apply the configuration)
>> VRRP VR 1 Priority Tracking# save                  (Save the configuration)
>> VRRP VR 1 Priority Tracking# ../vr 2               (Select virtual router 2 menu)
>> VRRP Virtual Router 2# ena                         (Enable the virtual router)
>> VRRP Virtual Router 2# vrid 2                     (Assign virtual router ID 2)
>> VRRP Virtual Router 2# if 2                       (To interface number 2)
>> VRRP Virtual Router 2# prio 101                   (Set the renter priority)
>> VRRP Virtual Router 2# addr 20.0.0.1              (Set IP address of virtual router)
>> VRRP Virtual Router 2# share dis                  (Disable sharing)
>> VRRP Virtual Router 2# track                      (Select Virtual Router Tracking Menu)
>> VRRP VR 2 Priority Tracking# ports ena             (Track VLAN switch ports)
>> VRRP VR 2 Priority Tracking# apply                 (Apply the configuration)
>> VRRP VR 2 Priority Tracking# save                  (Save the configuration)

```

7. Enable Server Load Balancing (SLB) on the first clean switch.

```

>> # /cfg/slb/on

```

8. Configure real servers for health checking VPN devices.

```

>> # /cfg/slb/real 1/ena                             (Enable slb for real server 1)
>> Real server 1 # rip 10.0.0.10                     (Assign IP address for real server 1)
>> Real server 1 # ../real 2/ena                     (Enable SLB for real server 2)
>> Real server 2 # rip 10.0.0.11                     (Assign IP address for real server 2)
>> Real server 2 # ../real 3/ena                     (Enable SLB for real server 3)
>> Real server 3 # rip 10.0.0.20                     (Assign IP address for real server 3)
>> Real server 3 # ../real 4/ena                     (Enable SLB for real server 4)
>> Real server 4 # rip 10.0.0.21                     (Assign IP address for real server 4)

```

9. Configure real server group 1, and add real servers 1, 2, 3, and 4 to the group.

```
>> # /cfg/slb/group 1                                (Configure real server group 1)
>> Real server group 1# metric hash                    (Select SLB hash metric for group 1)
>> Real server group 1# add 1                          (Add real servers 1-4 to group 1)
>> Real server group 1# add 2/add 3/add 4
```

10. Enable RTS on the necessary ports.

```
>> # /cfg/slb/port 26/rts ena                          (Enable Return to Sender on port 26)
>> # /cfg/slb/port 25/rts ena                          (Enable Return to Sender on port 25)
```

11. Enable filter processing on the server ports so that the responses from the real server is looked up in the VPN session table.

```
>> # /cfg/slb/port 1/filter ena
```

12. When dynamic routing protocols are not used, configure a gateway to the external router.

```
>> # cfg/ip/gw 1/addr 192.168.10.50
```

13. Apply and save the configuration, and reboot the switch.

```
>> # apply
>> # save
>> # /boot/reset
```

Configure the Second Clean-Side Application Switch-CB

1. Turn off bootp.

```
>> # /cfg/sys/bootp dis
```

2. Define and enable VLAN 2 for ports 25 and 26.

```
>> # /cfg/vlan 2/ena/def 25 26
```

3. Turn off Spanning Tree Protocol.

```
>> # /cfg/stp #/off
```

4. Define the clean-side IP interfaces.

Create one clean-side IP interface on a different subnet for each VPN device being load balanced.

```
>> # /cfg/ip/if 1/ena/mask 255.255.255.0/addr 30.0.0.11
>> # /cfg/ip/if 2/ena/mask 255.255.255.0/addr 20.0.0.20/v1 2
>> # /cfg/ip/if 3/ena/mask 255.255.255.255/addr 20.0.0.21/v1 2
```

5. Configure routes for each of the IP interfaces you configured in Step 4, using the VPN devices as gateways.

One static route is required for each VPN device being load balanced.

```
>> # /cfg/ip/route
>> # add 10.0.0.10 255.255.255.255 20.0.0.101 2
>> # add 10.0.0.11 255.255.255.255 20.0.0.102 3
>> # add 10.0.0.20 255.255.255.255 20.0.0.101 2
>> # add 10.0.0.21 255.255.255.255 20.0.0.102 3
```

6. Configure Virtual Router Redundancy Protocol (VRRP) for virtual routers 1 and 2.

```
>> # /cfg/vrrp/on
>> Virtual Router Redundancy Protocol# vr 1
>> VRRP Virtual Router 1# ena
>> VRRP Virtual Router 1# vrid 1
>> VRRP Virtual Router 1# if 1
>> VRRP Virtual Router 1# addr 30.0.0.50
>> VRRP Virtual Router 1# share dis
>> VRRP Virtual Router 1# track/vrs ena
>> VRRP Virtual Router 1 Priority Tracking# /cfg/vrrp/vr 2
>> VRRP Virtual Router 2# ena
>> VRRP Virtual Router 2# vrid 2
>> VRRP Virtual Router 2# if 2
>> VRRP Virtual Router 2# addr 20.0.0.1
>> VRRP Virtual Router 2# share dis
>> VRRP Virtual Router 2# track/ports ena
```

7. Enable SLB.

```
>> VRRP Virtual Router 2 Priority Tracking# /cfg/slb/on
```

8. Configure real servers for health checking VPN devices.

```
>> Layer 4# /cfg/slb/real 1/ena/rip 10.0.0.10
>> Real server 1# ../real 2/ena/rip 10.0.0.11
>> Real server 2# ../real 3/ena/rip 10.0.0.20
>> Real server 3# ../real 4/ena/rip 10.0.0.21
```

9. Enable the real server group.

```
>> Real server 4# ../group 1
>> Real server group 1# metric hash
>> Real server group 1# add 1/add 2/add 3/add 4
```

10. Enable RTS on the necessary ports.

```
>> Real server group 1# ../port 26/rts ena
>> SLB port 26# ../port 25/rts ena
```

11. Enable filter processing on the server ports so that the response from the real server will be looked up in VPN session table.

```
>> SLB port 25# ../port 1/filter ena
```

12. Apply and save the configuration, and reboot the switch.

```
>> SLB port 25# apply
>> SLB port 25# save
>> SLB port 25# /boot/reset
```

Configure the First Dirty-Side Application Switch-DA

1. Turn off BOOTP.

```
>> # /cfg/sys/bootp dis
```

2. Define and enable VLAN 2 for ports 25 and 26.

```
>> # /cfg/vlan 2/ena/def 25 26
```

3. Turn off STP.

```
>> # /cfg/stp/off
```

4. Configure IP interfaces 1, 2, and 3.

```
>> # /cfg/ip/if 1/ena/mask 255.255.255.0/addr 192.168.10.10
>> # /cfg/ip/if 2/ena/mask 255.255.255.0/addr 10.0.0.10/vl 2
>> # /cfg/ip/if 3/ena/mask 255.255.255.255/addr 10.0.0.11/vl 2
```

5. Define static routes for each of the IP interfaces you configured in Step 4, using the VPN devices as gateways.

One static route is required for each VPN device being load balanced.

```
>> # /cfg/ip/route
>> # add 20.0.0.10 255.255.255.255 10.0.0.101 2
>> # add 20.0.0.11 255.255.255.255 10.0.0.102 3
>> # add 20.0.0.20 255.255.255.255 10.0.0.101 2
>> # add 20.0.0.21 255.255.255.255 10.0.0.102 3
```

6. Configure VRRP for virtual routers 1 and 2.

```
>> # /cfg/vrrp/on
>> Virtual Router Redundancy Protocol# /cfg/vrrp/vr 1
>> VRRP Virtual Router 1# ena
>> VRRP Virtual Router 1# vrid 1
>> VRRP Virtual Router 1# if 1
>> VRRP Virtual Router 1# prio 101
>> VRRP Virtual Router 1# addr 192.168.10.50
>> VRRP Virtual Router 1# share dis
>> VRRP Virtual Router 1# track
>> VRRP Virtual Router 1 Priority Tracking# vrs ena
>> VRRP Virtual Router 1 Priority Tracking# ports ena
>> VRRP Virtual Router 1 Priority Tracking# /cfg/vrrp/vr 2
>> VRRP Virtual Router 2# ena
>> VRRP Virtual Router 2# vrid 2
>> VRRP Virtual Router 2# if 2
>> VRRP Virtual Router 2# prio 101
>> VRRP Virtual Router 2# addr 10.0.0.1
>> VRRP Virtual Router 2# share dis
>> VRRP Virtual Router 2# track
>> VRRP Virtual Router 2 Priority Tracking# vrs ena
>> VRRP Virtual Router 2 Priority Tracking# ports ena
```

7. Enable SLB.

```
>> VRRP Virtual Router 1 Priority Tracking# /cfg/slb/on
```

8. Configure real servers for health-checking VPN devices.

```
>> Layer 4# real 1/ena/rip 20.0.0.10
>> Real server 1# ../real 2/ena/rip 20.0.0.11
>> Real server 2# ../real 3/ena/rip 20.0.0.20
>> Real server 3# ../real 4/ena/rip 20.0.0.21
```

9. Enable the real server group.

```
>> Real server 1# ../group 1
>> Real server group 1# metric hash
>> Real server group 1# add 1/add 2/add 3/add 4
```

10. Configure the filters to allow local subnet traffic on the dirty side of the VPN device to reach the VPN device interfaces.

```
>> # ../filt 100
>> # ena
>> # sip any
>> # dip 192.168.10.0/dmask 255.255.255.0
>> # action allow
>> # ../filt 110
>> # ena
>> # sip any
>> # dip 224.0.0.0/dmask 255.0.0.0
>> # action allow
```

11. Create the redirection filter and enable firewall load balancing.

This filter will redirect inbound traffic, redirecting it among the defined real servers in the group.

```
>> # ../filt 2048
>> # ena
>> # sip any
>> # dip any
>> # action redir
>> # ../filt 2048/adv
>> # fwlb ena
```


12. Create a filter to allow the management firewall (Policy Server) to reach the VPN firewall.

```
>> # ../filt 120 ena
>> # sip 192.168.10.120
>> # smask 255.255.255.255
>> # dip 10.0.0.0
>> # dmask 255.255.255.0
```

13. Add filters to the ingress port.

```
>> # ../port 1
>> # filt ena
>> # add 100/add 110/add 2048
```

14. Apply and save the configuration, and reboot the switch.

```
>> # apply
>> # save
>> # /boot/reset
```

Configure the Second Dirty-Side Application Switch-DB

1. Turn off BOOTP.

```
>> # /cfg/sys/bootp dis
```

2. Define and enable VLAN 2 for ports 25 and 26.

```
>> # /cfg/vlan 2/ena/def 25 26
```

3. Turn off STP.

```
>> # /cfg/stp/off
```

4. Configure IP interfaces 1, 2, and 3.

```
>> # /cfg/ip/if 1/ena/mask 255.255.255.0/addr 192.168.10.11
>> # /cfg/ip/if 2/ena/mask 255.255.255.0/addr 10.0.0.20/vl 2
>> # /cfg/ip/if 3/ena/mask 255.255.255.255/addr 10.0.0.21/vl 2
```

5. Configure routes for each of the IP interfaces you configured in Step 4.

```
>> # /cfg/ip/route
>> # add 20.0.0.10 255.255.255.255 10.0.0.101 2
>> # add 20.0.0.11 255.255.255.255 10.0.0.102 3
>> # add 20.0.0.20 255.255.255.255 10.0.0.101 2
>> # add 20.0.0.21 255.255.255.255 10.0.0.102 3
```

6. Configure VRRP for virtual routers 1 and 2.

```
>> # /cfg/vrrp/on
>> # /cfg/vrrp/vr 1
>> # ena
>> # vrid 1
>> # if 1
>> # addr 192.168.10.50
>> # share dis
>> # track
>> # vrs ena
>> # ports ena
>> # /cfg/vrrp/vr 2
>> # ena
>> # vrid 2
>> # if 2
>> # addr 10.0.0.1
>> # share dis
>> # track
>> # vrs ena
>> # ports ena
```

7. Enable SLB.

```
>> # /cfg/slb/on
```

8. Configure real servers for health checking VPN devices.

```
>> # /cfg/slb/real 1/ena/rip 20.0.0.10
>> # /cfg/slb/real 2/ena/rip 20.0.0.11
>> # /cfg/slb/real 3/ena/rip 20.0.0.20
>> # /cfg/slb/real 4/ena/rip 20.0.0.21
```

9. Enable the real server group, and place real servers 1-4 into the real server group.

```
>> # /cfg/slb/group 1
>> # metric hash
>> # add 1/add 2/add 3/add 4
```

10. Configure the filters to allow local subnet traffic on the dirty side of the VPN device to reach the VPN device interfaces.

```
>> # /cfg/slb/filt 100
>> # ena
>> # sip any
>> # dip 192.168.10.0/dmask 255.255.255.0
>> # ../filt 110
>> # ena
>> # sip any
>> # dip 224.0.0.0/dmask 255.0.0.0
```

11. Create the redirection filter and enable firewall load balancing.

This filter will redirect inbound traffic, among the defined real servers in the group.

```
>> # /cfg/slb/filt 2048
>> # ena
>> # sip any
>> # dip any
>> # proto any
>> # action redir
>> # ../filt 2048/adv
>> # fwlb ena
```

12. Add filters to the ingress port.

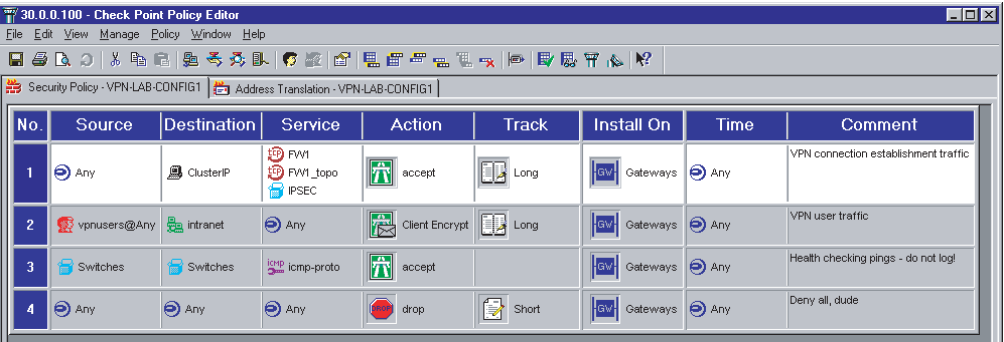
```
>> # /cfg/slb/port 1
>> # filt ena
>> # add 100/add 110/add 2048
```

13. Apply and save the configuration and reboot the switch.

```
>> # apply
>> # save
>> # /boot/reset
```

Test Configurations and General Topology

The application switches should be able to perform health checks to each other and all switches should see four real servers (see [Figure 16-3](#)).



The screenshot shows the 'Check Point Policy Editor' window. It displays a table of security rules for 'VPN-LAB-CONFIG1'. The table has columns for No., Source, Destination, Service, Action, Track, Install On, Time, and Comment. There are four rules listed:

No.	Source	Destination	Service	Action	Track	Install On	Time	Comment
1	Any	ClusterIP	FW1, FW1_topo, IPSEC	accept	Long	Gateways	Any	VPN connection establishment traffic
2	vpnusers@Any	intranet	Any	Client Encrypt	Long	Gateways	Any	VPN user traffic
3	Switches	Switches	icmp-proto	accept		Gateways	Any	Health checking pings - do not log!
4	Any	Any	Any	drop	Short	Gateways	Any	Deny all, dude

Figure 16-3 Checkpoint Rules for Both VPN Devices as Seen in the Policy Editor

- 1. **Disconnect the cables (cause failures) to change the available servers that are up.**

```
>> # /info/slb/dump
```

(Verify which servers are up)

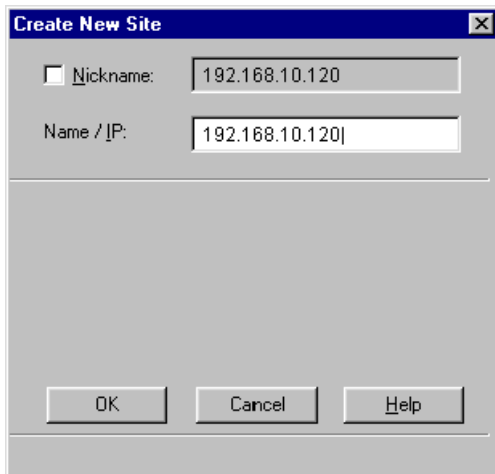
This should change the VRRP preferences. You can view VRRP preferences using the CLI command `/info/vrrp`.

- 2. **Use the Checkpoint Log Viewer to watch for accepted and dropped traffic. In the tool bar above, click on Window, then Log Viewer.**

NOTE – To help simplify the logs, the health checks are *not* logged.

Test the VPN

1. Launch the SecuRemote client on the dirty side of the network.
2. Add a new site.

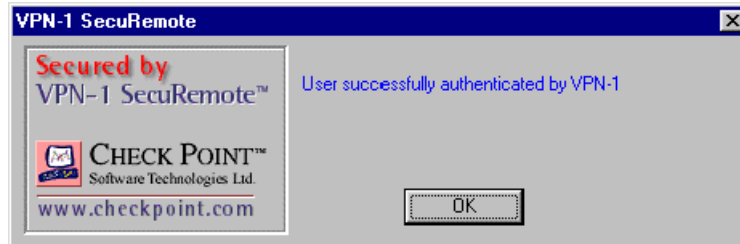


3. Enter the policy server IP address: 192.168.10.120. You have the option of adding a nickname.
4. Launch a browser (such as Netscape or Internet Explorer) and go to <http://30.0.0.100>
5. You will be prompted for authentication.

Enter **vpnuser** for user name and **alteon** for the password.



6. A message is displayed verifying that you were authenticated.



7. Browse to the Web site.

If there are other services running on other servers in the internal network, you should be able to reach those services. All traffic traveling over the VPN is decrypted at the VPN device. You can verify which VPN device is being used by looking at the Log Viewer. You should also be able to see the client authentication as well as the decrypted traffic.

To verify that the FWLB and hash metric is working correctly on the dirty-side switches (that is, hashed on client IP address/destination IP address), configure your current client with an IP address one higher (or lower) in the last octet, and try to reestablish the VPN connection. Or, add another PC on the dirty side and connect to it.

NOTE – When many clients are coming from *behind* a VPN gateway (for example, not using the SecuRemote clients but using a VPN 1 Gateway or other compatible VPN Gateway), you will *not* see load balancing across those clients. Each SecuRemote client will be treated differently, but each VPN 1 Gateway will be treated as one client each (that is, one Client IP address). VPN Device 1 and VPN Device 2 belong to one cluster IP.



CHAPTER 17

Content Intelligent Switching

This chapter discusses advanced load balancing solutions utilizing Layer 7 content switching. Inspecting HTTP headers, examining content identifiers such as URLs and cookies, and parsing content requests are discussed in the following topics:

- “Overview” on page 416
- “Content Intelligent Server Load Balancing” on page 419
- “Content Intelligent Cache Redirection” on page 442
- “Exclusionary String Matching for Real Servers” on page 461
- “Regular Expression Matching” on page 463
- “Content Precedence Lookup” on page 465
- “Layer 7 Deny Filters” on page 468

NOTE – For all content-intelligent switching features enable Direct Access Mode (DAM) or configure proxy IP addresses.

Overview

Alteon Application Switches perform content intelligent switching by processing numerous tasks for each incoming session, including connection setup, traffic parsing, applying server selection algorithms, splicing connections and translating session addresses, metering and controlling server bandwidth usage, processing traffic filters, collecting statistics, and so on. [Figure 17-1](#) illustrates the process of content intelligent switching in the application switch.

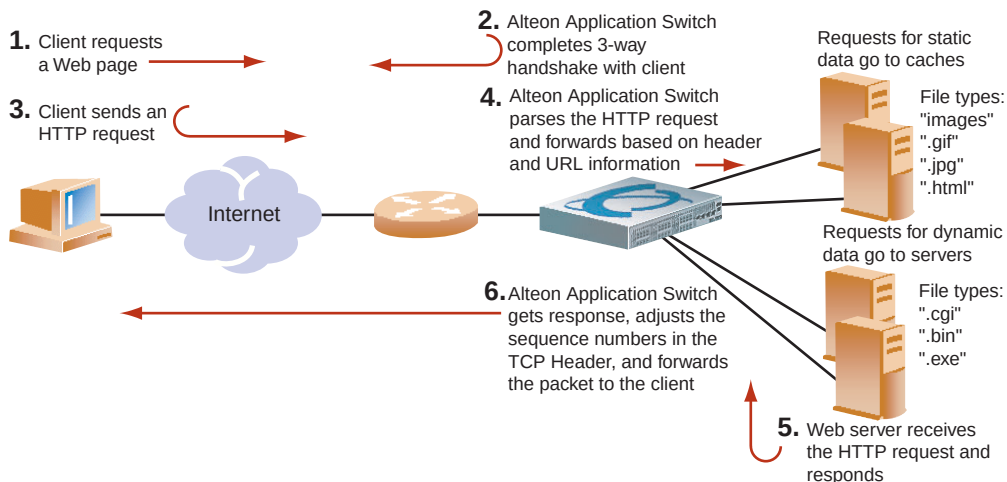


Figure 17-1 Content Intelligent Load Balancing Example

Parsing Content

Examining session content places heavier demands upon the application switch than examining TCP/IP headers for the following reasons:

- Content is non-deterministic. Content identifiers such as URLs and cookies can be of varying lengths and can appear at unpredictable locations within a request. Scanning session traffic for a specific string is far more processor-intensive than looking at a known location in a session for a specific number of bytes.
- To parse a content request, the application switch must temporarily terminate the TCP connection from the client. This temporary termination is called a *delayed binding*. While the connection is suspended, the application switch acknowledges the client connection on behalf of the server, buffers and examines the client request, and finally opens a connection to an appropriate server based on the requested content.

For more information on delayed binding, see [“Delayed Binding” on page 167](#).

- Delayed binding causes two independent TCP connections to span a Web session: one from the client to the application switch and the second from the application switch to the selected server. The application switch must modify the TCP header, including performing TCP sequence number translation and recalculating checksums on every packet that travels between the client and the server, for the duration of the session. This function, known as *TCP connection splicing*, heavily tasks a application switch, particularly when the switch must process thousands of these sessions simultaneously.
- In addition to real-time traffic and connection processing, a content intelligent application switch must monitor servers to ensure that requests are forwarded to the best-performing and healthiest servers. This monitoring involves more than simple ICMP or TCP connection tests, as servers can continue to process network protocols while failing to retrieve content.
- If content is segregated in different servers or server farms, the application switch must provide a flexible, user-customizable mechanism allowing a relevant set of application and content tests to be applied to each server or server farm.

In addition to implementing content intelligent switching, the switch periodically performs background functions such as updating network topology, measuring server performance, and health checking for servers, applications, and server sites.

HTTP Header Inspection

Content intelligent switching is performed by inspecting HTTP headers. HTTP headers include additional information about requests and responses. The HTTP 1.1 specification defines a total of 46 headers. For cache redirection, at any given time one HTTP header is supported globally for the entire switch.

HTTP headers can be general, request, response, or entity headers. General headers may exist in both requests and responses. Requests and response headers are specific only to requests and responses, respectively. Entity headers describe the content of the request body or the content of the response body.

Each HTTP header field consists of a name, followed immediately by a colon (:), a single space character, and the field value. Field names are case-insensitive. Header fields can be extended over multiple lines by preceding each extra line with at least one space.

Some customer applications of HTTP header inspection are listed below:

- Redirection based on domain name
- Cachability based on domain name
- Virtual hosting
- Redirection based on browser type
- Cookie-based preferential redirection

Buffering Content with Multiple Frames

To handle the overall length of HTTP headers, including request headers containing multiple cookies, and the Maximum Segment Size (MSS) of dial-up connections, Alteon OS software provides the following support:

- HTTP GET Request Headers

NOTE – In addition to the URL path, which generally is less than 300 bytes, the HTTP GET requests also include general headers and request headers.

- ☐ Parsing GET requests to match URL path and HTTP header beyond the first frame while performing delayed binding
- ☐ Processing multiple frames from a single HTTP GET request, using a TCP stack on the switch
- HTTP Cookie Request Headers
 - Buffering a maximum of 4500 bytes for a single GET request across multiple frames. A single GET request can include multiple cookies.

Content Intelligent Server Load Balancing

Alteon OS allows you to load balance HTTP requests based on different HTTP header information, such as “Cookie:” header for persistent load balancing, “Host:” header for virtual hosting, or “User-Agent” for browser-smart load balancing.

- [URL-Based Server Load Balancing](#) on this page
- [“Virtual Hosting”](#) on page 424
- [“Cookie-Based Preferential Load Balancing”](#) on page 427
- [“URL Hashing for Server Load Balancing”](#) on page 431
- [“Header Hash Load Balancing”](#) on page 433
- [“DNS Load Balancing”](#) on page 434
- [“RTSP Load Balancing”](#) on page 437

URL-Based Server Load Balancing

URL-based SLB allows you to optimize resource access and server performance. Content dispersion can be optimized by making load-balancing decisions on the entire path and filename of each URL.

NOTE – Both HTTP 1.0 and HTTP 1.1 requests are supported.

For URL matching you can configure up to 512 strings comprised of 40 bytes each. Each URL request is then examined against the URL strings defined for each real server. URL requests are load balanced among multiple servers matching the URL, according to the load balancing metric configured for the real server group (`leastConns` is the default).

In [Figure 17-2](#), the following criteria are specified for content load balancing:

- Requests with “.cgi” in the URL are forwarded to real servers 3 and 4.
- Requests with the string “images” in the URL are sent to real servers 1 and 2.
- Requests with URLs starting with “/product:” are sent to real servers 2, 3, and 5.

Requests containing URLs with anything else are sent to real servers 1, 2, 3, and 4. These servers have been defined with the “any” string.

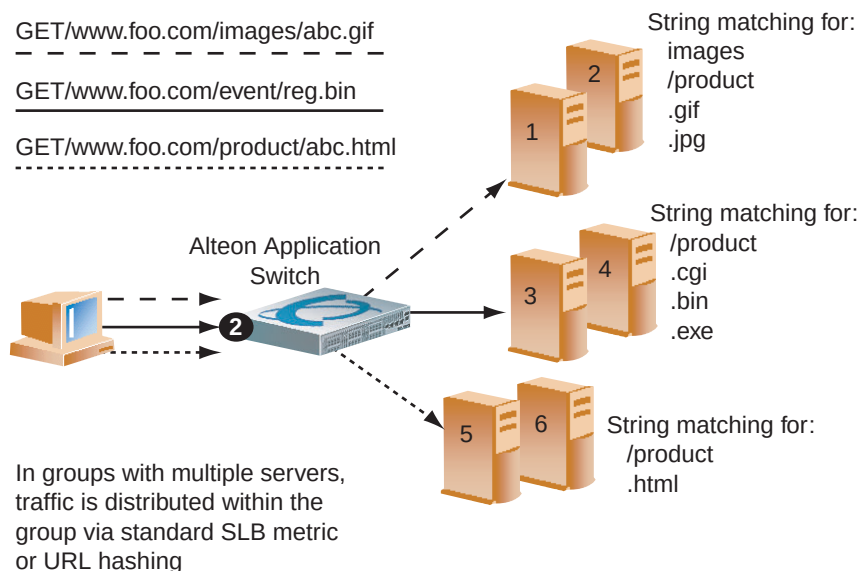


Figure 17-2 URL-Based Server Load Balancing

Configuring URL-Based Server Load Balancing

To configure URL-based SLB, perform the following steps:

- 1. Before you can configure URL-based load balancing, ensure that the switch has already been configured for basic SLB with the following tasks:**

NOTE – When URL-based SLB is used in an active/active redundant setup, use a proxy IP address instead of Direct Access Mode (DAM) to enable the URL parsing feature.

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Define a real server group and set up health checks for the group.
- Define a virtual server on virtual port 80 (HTTP), and assign the real server group to service it.
- Enable SLB on the switch.
- Enable client processing on the port connected to the clients.

For information on how to configure your network for SLB, see [“Server Load Balancing” on page 137](#).

2. Define the string(s) to be used for URL load balancing.

```
>> # /cfg/slb/layer7/slb/add|rem <string>
```

- **add:** Add string or a path.
- **rem:** Remove string or a path.

A default string “any” indicates that the particular server can handle all URL or cache requests. Refer to the following examples given below:

Example 1: String with the Forward Slash (/)

A string that starts with a forward slash (/), such as “/images,” indicates that the server will process requests that start out with the “/images” string only.

For example, with the “/images” string, the server will handle these requests:

```
/images/product/b.gif
/images/company/a.gif
/images/testing/c.jpg
```

The server will *not* handle these requests:

```
/company/images/b.gif
/product/images/c.gif
/testing/images/a.gif
```

Example 2: String without the Forward Slash (/)

A string that does not start out with a forward slash (/) indicates that the server will process any requests that contain the defined string. For example, with the “images” string, the server will process these requests:

```
/images/product/b.gif
/images/company/a.gif
/images/testing/c.jpg
/company/images/b.gif
/product/images/c.gif
/testing/images/a.gif
```

Example 3: String with the Forward Slash (/) Only

If a server is configured with the load balance string (/) only, it will only handle requests to the root directory. For example, the server will handle any files in the `root` directory:

```
/
/index.htm
/default.asp
/index.shtm
```

- 3. **Apply and save your configuration changes.**
- 4. **Identify the defined string IDs.**

```
>> # /cfg/slb/layer7/slb/cur
```

For easy configuration and identification, each defined string is assigned an ID number, as shown in the following example:

ID	SLB String
1	any
2	.gif
3	/sales
4	/xitami
5	/manual
6	.jpg

- 5. **Configure one or more real servers to support URL-based load balancing.**
- 6. **Add the defined string(s) to the real server using the following command:**

```
>> # /cfg/slb/real 2/layer7/addlb <ID>
```

where *ID* is the identification number of the defined string.

NOTE – If you don't add a defined string (or add the defined string “any”) the server will handle any request.

A server can have multiple defined strings. For example:

- “/images”
- “/sales”
- “.gif”

With these defined strings, this particular server can handle requests that start with “/images” or “/sales” and any requests that contain “.gif”.

- 7. **Enable SLB on the switch.**

```
>> # /cfg/slb/on (Turn SLB on)
```

8. **Enable DAM on the switch or configure proxy IP addresses and enable proxy on the client port.**

DAM and proxy IPs allow you to perform port mapping for URL load balancing.

■ Enable DAM

```
>> # /cfg/slb/adv/direct ena
```

■ Configure a proxy IP address and enable proxy on the client port

```
>> # /cfg/slb/direct dis
>> # /cfg/slb/pip
>> Proxy IP Address# pip1 12.12.12.12
>> # /cfg/slb/port 2/proxy ena
```

For more information on proxy IP addresses, see [“Proxy IP Addresses” on page 156](#).

9. **Enable URL-based SLB on the virtual server(s).**

```
>> # /cfg/slb/virt <virtual server number>/service 80/httpslb urlslb
```

Statistics for URL-Based Server Load Balancing

To show the number of hits to the SLB or cache server, use this command:

```
>> # /stats/slb/layer7/str
```

Sample Statistics:

ID	SLB String	Hits
1	any	73881
2	.gif	0
3	/sales	0
4	/xitami	162102
5	/manual	0
6	.jpg	0

Virtual Hosting

Alteon OS allows individuals and companies to have a presence on the Internet in the form of a dedicated Web site address. For example, you can have a “www.site-a.com” and “www.site-b.com” instead of “www.hostsite.com/site-a” and “www.hostsite.com/site-b.”

Service providers, on the other hand, do not want to deplete the pool of unique IP addresses by dedicating an individual IP address for each home page they host. By supporting an extension in HTTP 1.1 to include the host header, Alteon OS enables service providers to create a single virtual server IP address to host multiple Web sites per customer, each with their own host name.

NOTE – For SLB, one HTTP header is supported per virtual server.

The following list provides more detail on virtual hosting with configuration information.

- An HTTP/1.0 request sent to an origin server (*not* a proxy server) is a partial URL instead of a full URL.

An example of the request that the origin server would see as follows:

```
GET /products/2424/ HTTP/1.0
User-agent: Mozilla/3.0
Accept: text/html, image/gif, image/jpeg
```

The GET request does not include the host name. From the TCP/IP headers, the origin server knows the requests host name, port number, and protocol.

- With the extension to HTTP/1.1 to include the HTTP HOST: header, the above request to retrieve the URL “/www.nortelnetworks.com/ products/2424” would look like this:

```
GET /products/2424/ HTTP/1.1
Host: www.nortelnetworks.com
User-agent: Mozilla/3.0
Accept: text/html, image/gif, image/jpeg
```

The Host: header carries the hostname used to generate the IP address of the site.

- Based on the Host: header, the switch will forward the request to servers representing different customers’ Web sites.
- The network administrator needs to define a domain name as part of the 128 supported URL strings.
- The switch performs string matching; that is, the string “nortelnetworks.com” or “www.nortelnetworks.com” will match “www.nortelnetworks.com.”

Virtual Hosting Configuration Overview

The sequence of events for configuring virtual hosting based on HTTP Host: headers is described below:

1. **The network administrator defines a domain name as part of the 128 supported URL strings.**

Both domain names “www.company-a.com” and “www.company-b.com” resolve to the same IP address. In this example, the IP address is for a virtual server on the switch.

2. **“www.company-a.com” and “www.company-b.com” are defined as URL strings.**

3. **Server Group 1 is configured with Servers 1 through 8.**

Servers 1 through 4 belong to “www.company-a.com” and Servers 5 through 8 belong to “www.company-b.com.”

4. **The network administrator assigns string “www.company-a.com” to Servers 1 through 4 and string “www.company-b.com” to Servers 5 through 8.**

5. **The application switch inspects the HTTP host header in requests received from the client.**

- If the host header is “www.company-a.com,” the switch directs requests to one of the Servers 1 through 4.
- If the host header is “www.company-b.com,” the switch directs requests to one of the Servers 5 through 8.

Configuring the “Host” Header for Virtual Hosting

To support virtual hosting, configure the switch for Host header-based load balancing with the following procedure:

- Before you can configure host header-based server load balancing, ensure that the switch has already been configured for basic SLB:**

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Assign servers to real server groups.
- Define virtual servers and services.

For information on how to configure your network for server load balancing, see [“Server Load Balancing” on page 137](#).

- Turn on URL parsing for the virtual server for virtual hosting.**

```
>> # /cfg/slb/virt 1                                (Select the virtual IP for host
                                                         header-based SLB)
>> Virtual Server 1 # service 80                    (Select the HTTP service)
>> Virtual Server 1 http Service # httpslb host
```

- Define the host names.**

```
>> # /cfg/slb/layer7/slb/add "www.customer1.com"
>> Server Loadbalance Resource# add "www.customer2.com"
>> Server Loadbalance Resource# add "www.customer3.com"
```

- Configure the real server(s) to handle the appropriate load balancing string(s).**

To add a defined string:

```
>> # /cfg/slb/real 2                                (Select the real server)
>> Real Server 2 # Layer7
>> Real Server 2 Layer 7 Commands # addlb <ID>(Specify the string ID)
```

where *ID* is the identification number of the defined string.

NOTE – If you don't add a defined string (or add the defined string “any”), the server will handle any request.

Cookie-Based Preferential Load Balancing

Cookies can be used to provide preferential services for customers, ensuring that certain users are offered better access to resources than other users when site resources are scarce. For example, a Web server could authenticate a user via a password and then set cookies to identify them as “Gold,” “Silver,” or “Bronze” customers. Using cookies, you can distinguish individuals or groups of users and place them into groups or communities that get redirected to better resources and receive better services than all other users.

NOTE – [Cookie-based persistent load balancing is described in Chapter 18, “Persistence.”](#)

Cookie-based preferential services enable the following support:

- Redirect higher priority users to a larger server or server group.
- Identify a user group and redirect them to a particular server.
- Serve content based on user identity.
- Prioritize access to scarce resources on a Web site.
- Provide better services to repeat customers, based on access count.

Clients that receive preferential service can be distinguished from other users by one of the following methods:

- **Individual User**

Specific individual user could be distinguished by IP address, login authentication, or permanent HTTP cookie.

- **User Communities**

Some set of users, such as “Premium Users” for service providers who pay higher membership fees than “Normal Users” could be identified by source address range, login authentication, or permanent HTTP cookie.

- **Applications**

Users could be identified by the specific application they are using. For example, priority can be given to HTTPS traffic that is performing credit card transactions versus HTTP browsing traffic.

- **Content**

Users could be identified by the specific content they are accessing.

Based on one or more of the criteria above, you can load balance requests to different server groups.

Configuring Cookie-Based Preferential Load Balancing

To configure cookie-based preferential load balancing, perform the following procedure.

- 1. Before you can configure header-based load balancing, ensure that the switch has already been configured for basic SLB with the following tasks:**

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Assign servers to real server groups.
- Define virtual servers and services.

For information on how to configure your network for SLB, see [Chapter 9, “Server Load Balancing”](#).

- 2. Turn on URL parsing for the virtual server.**

```
>> # /cfg/slb/virt 1
>> Virtual Server 1 # service 80
>> Virtual Server 1 http Service # httpslb cookie
Enter Cookie Name: sid
Enter the starting point of the Cookie value [1-64]: 1
Enter the number of bytes to extract [1-64]: 6
Look for Cookie in URI [e|d]: d
```

where

sid = cookie name

1 = offset (the starting position of the value to be used for hashing)

6 = length (the number of bytes in the cookie value)

d = looks for the cookie in the cookie header instead of the URI (disables searching for cookie in the URI)

- 3. Define the cookie values.**

```
>> # /cfg/slb/layer7/slb/add "Gold"
>> # add "Silver"
>> # add "Bronze"
```

Since a session cookie does not exist in the first request of an HTTP session, a default server or “any” server is needed to assign cookies to a “None” cookie HTTP request.

Example:

- Real Server 1: “Gold” handles gold requests.
- Real Server 2: “Silver” handles silver request.
- Real Server 3: “Bronze” handles bronze request.
- Real Server 4: “any” handles any request that does not have a cookie or matching cookie.

With servers defined to handle the requests listed above, here is what happens:

- Request 1 comes in with no cookie; it is forwarded to Real Server 4 to get cookie assigned.
- Request 2 comes in with “Gold” cookie; it will be forwarded to Real Server 1.
- Request 3 comes in with “Silver” cookie; it will be forwarded to Real Server 2.
- Request 4 comes in with “Bronze” cookie; it will be forwarded to Real Server 3.
- Request 5 comes in with “Titanium” cookie; it will be forwarded to Real Server 4, since it does not have an exact cookie match (matches with “any” configured at Real Server 4).

4. Configure the real server(s) to handle the appropriate load balance string(s).

To add a defined string:

```
>> # /cfg/slb/real 2/layer7/addlb <ID>
```

where *ID* is the identification number of the defined string.

NOTE – If you don't add a defined string (or add the defined string “any”), the server will handle any request.

5. Enable DAM on the switch or configure proxy IP addresses and enable proxy on the client port.

To use cookie-based preferential load balancing without DAM, you must configure proxy IP addresses.

Enable proxy load balancing on the port used for cookie-based preferential load balancing. If Virtual Matrix Architecture (VMA) is enabled on the switch, you can choose to configure the remaining ports with proxy IP disabled.

Browser-Smart Load Balancing

HTTP requests can be directed to different servers based on browser type by inspecting the “User-Agent” header. For example,

```
GET /products/2424/ HTTP/1.0
User-agent: Mozilla/3.0
Accept: text/html, image/gif, image/jpeg
```

To allow the switch to perform browser-smart load balancing, perform the following procedure.

1. **Before you can configure browser-based load balancing, ensure that the switch has already been configured for basic SLB with the following tasks:**
 - Assign an IP address to each of the real servers in the server pool.
 - Define an IP interface on the switch.
 - Define each real server.
 - Assign servers to real server groups.
 - Define virtual servers and services.
2. **Turn on URL parsing for the virtual server for “User-Agent:” header.**

```
>> # /cfg/slb/virt 1/service 80/httpslb browser
```

3. **Define the host names.**

```
>> # /cfg/slb/layer7/slb/add "Mozilla"
>> Server Loadbalance Resource# add "Internet Explorer"
>> Server Loadbalance Resource# add "Netscape"
```

4. **Configure the real server(s) to handle the appropriate load balancing string(s).**

NOTE – If you don't add a defined string (or add the defined string “any”), the server will handle any request.

Use the following command to add a defined string:

```
>> # /cfg/slb/real 2/layer7/addlb <ID>
```

where *ID* is the identification number of the defined string.

URL Hashing for Server Load Balancing

By default, hashing algorithms use the IP source address and/or IP destination address (depending on the application area) to determine content location. The default hashing algorithm for SLB is the IP source address. By enabling URL hashing, requests going to the same page of an origin server are redirected to the same real server or cache server.

Load Balancing Nontransparent Caches

You can deploy a cluster of non-transparent caches and use the virtual server to load balance requests to the cache servers. The client's browser is configured to send Web requests to a non-transparent cache (the IP address of the configured virtual server).

If hash is selected as the load-balancing algorithm, the switch hashes the source IP address to select the server for SLB. Under this condition, the switch may not send requests for the same origin server to the same proxy cache server. For example, requests made from a client to “http://www.nortelnetworks.com/products” from different clients may get sent to different caches.

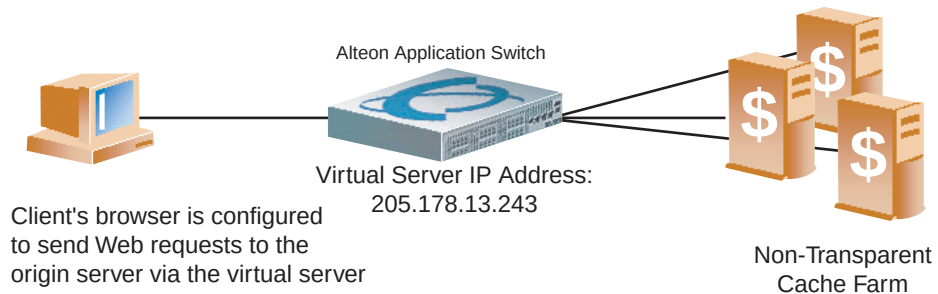


Figure 17-3 Using URL hashing to Load Balance Nontransparent Caches

Configuring URL Hashing

You can direct the same URL request to the same cache or proxy server by using a virtual server IP address to load balance proxy requests. By configuring `hash` or `minmisses` as the metric, the switch uses the number of bytes in the URI to calculate the hash key.

If the host field exists and the switch is configured to look into the Host: header, the switch uses the Host: header field to calculate the hash key.

To configure URL hashing, perform the following procedure:

1. Before you can configure URL hashing, ensure that the switch has already been configured for basic SLB with the following tasks:

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Assign servers to real server groups.
- Define virtual servers and services.
- Configure load-balancing algorithm for hash or minmiss.
- Enable SLB.

For information on how to configure your network for SLB, see [Chapter 9, “Server Load Balancing.”](#)

- Define server port and client port.

2. Enable URL hashing.

```
>> # /cfg/slb/virt 1
>> Virtual Server 1 # service 80
>> Virtual Server 1 http Service # httpslb urlhash
Enter new hash length [1-255]: 25
```

Hashing is based on the URL, including the HTTP Host: header (if present), up to a maximum of 255 bytes.

3. Set the metric for the real server group to minmisses or hash.

```
>> # /cfg/slb/group 1/metric <hash|minmisses>
```


Header Hash Load Balancing

Alteon OS allows you to hash on *any* selected HTTP header. To configure the application switch for load balancing based on header hash, perform the following procedure:

1. Ensure that the switch has already been configured for basic SLB:

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Assign servers to real server groups.
- Define virtual servers and services.

For information on how to configure your network for SLB, see [Chapter 9, “Server Load Balancing.”](#)

2. Enable header hashing.

```
>> # /cfg/slb/virt 1
>> Virtual Server 1 # service 80
>> Virtual Server 1 http Service # httpslb headerhash
Select Operation: none
Enter new HTTP header name: User-agent
Enter new hash length [1-255]: 25
```

3. Set the metric for the real server group to minmisses or hash.

```
>> # /cfg/slb/group 1/metric <hash|minmisses>
```

DNS Load Balancing

The Internet name registry has become so large that a single server cannot keep track of all the entries. This is resolved by splitting the registry and saving it on different servers.

If you have large DNS server farms, Alteon OS allows you to load balance traffic based on DNS names. To load balance DNS names, the host name is extracted from the query, processed by the regular expressions engine, and the request is sent to the appropriate real server.

For example, [Figure 17-4](#) shows a DNS server farm load balancing DNS queries based on DNS names. Requests with DNS names beginning with A through G are sent to Server 1; DNS names beginning with H through M are sent to Server 2; DNS names beginning with N through T are sent to Server 3; DNS names beginning with U through Z are sent to Server 4.

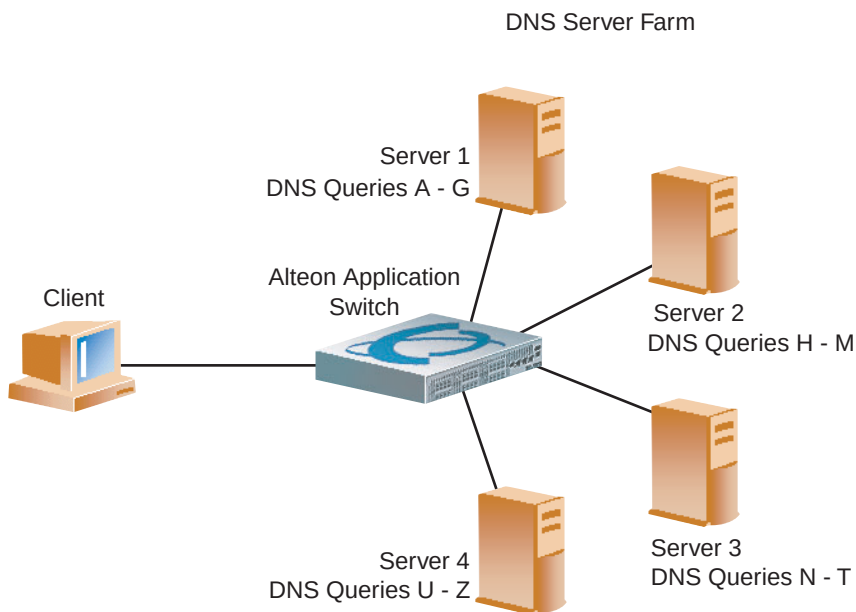


Figure 17-4 Load Balancing DNS Queries

To configure the switch for DNS load balancing, perform the following procedure:

1. Before you can configure DNS load balancing, ensure that the switch has already been configured for basic SLB with the following tasks:

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server (DNS server address).
- Assign servers to real server groups.
- Define virtual servers and services.
- Enable SLB.

For information on how to configure your network for SLB, see [Chapter 9, “Server Load Balancing.”](#)

- Define server port and client port.

2. Enable DNS load balancing.

```
>> # /cfg/slb/virt 1                               (Select the virtual server)
>> Virtual Server 1 # service 53                     (Select the DNS service)
>> Virtual Server 1 DNS Service # dnsslb ena(Enable DNS SLB)
```

3. Enable delayed binding.

```
>> Virtual Server 1 DNS Service# dbind ena
```

4. Define the host names.

```
>> # /cfg/slb/layer7/slb/add www.[abcdefg]*.com
>> Server Loadbalance Resource# add www.[hijklm]*.com
>> Server Loadbalance Resource# add www.[nopqrst]*.com
>> Server Loadbalance Resource# add www.[vwxyz]*.com
```

5. Apply and save your configuration changes.

6. Identify the defined string IDs.

```
>> # /cfg/slb/layer7/slb/cur
```

For easy configuration and identification, each defined string has an ID attached, as shown in the following example:

ID	SLB String
1	any, cont 1024
2	www.[abcdefg]*.com, cont 1024
3	www.[hijklm]*.com, cont 1024
4	www.[nopqrst]*.com, cont 1024
5	www.[uvwxyz]*.com, cont 1024

7. Add the defined string IDs to the real server using the following command:

```
>> # /cfg/slb/real 1/layer7/addlb 2  
>> # /cfg/slb/real 2/layer7/addlb 3  
>> # /cfg/slb/real 3/layer7/addlb 4
```

NOTE – If you don't add a defined string (or add the defined string “any”) the server will handle any request.

RTSP Load Balancing

Alteon OS supports RTSP load balancing based on URL hash metric or string matching to load balance media servers that contain multimedia presentations. Because multimedia presentations consume a large amount of Internet bandwidth, and their correct presentation depends upon the real time delivery of the data over the Internet, several media servers contain the same multimedia data.

For more conceptual information on RTSP, refer to [“Real Time Streaming Protocol SLB” on page 176](#).

[Figure 17-5](#) shows two groups of media servers: Group 1 is configured for URL hashing and group 2 is configured for string matching. The media servers are cache servers configured in reverse proxy mode.

URL Hash

Use the URL hash metric to maximize client requests to hash to the same media server. The original servers push the content to the cache servers ahead of time. For example in [Figure 17-5](#), an ISP is hosting audio-video files for NortelNews on media servers 1, 2, 3, and 4. The domain name nortelnews.com associated with the virtual IP address 120.10.10.10 is configured for URL hash.

The first request for `http://nortelnews.com/saleswebcast.rm` hashes to media server 1. Subsequent requests for `http://nortelnews.com/saleswebcast.rm` from other clients or from client 1 will hash to the same server 1. Similarly, another request for `http://nortelnews.com/marketingwebcast.rm` may hash to media server 2, provided saleswebcast and marketingwebcast media files are located in the origin servers.

Typically, a set of related files (audio, video, and text) of a presentation are usually placed under the same directory (called container directory). Alteon OS URL hashing ensures that the entire container is cached in a single cache by using the entire URL to compute the hash value and omitting the extension (for example, *.ram*, *.rm*, *.smil*) occurring at the end of the URL.

String Matching

Use the string matching option to populate the RTSP servers with content-specific information. For example, you have clients accessing audio-video files on StarWars1 and clients accessing audio-video files on StarWars2. You can host the StarWars1 media files on media servers 5 and 6 and host StarWars2 media files on media servers 7 and 8.

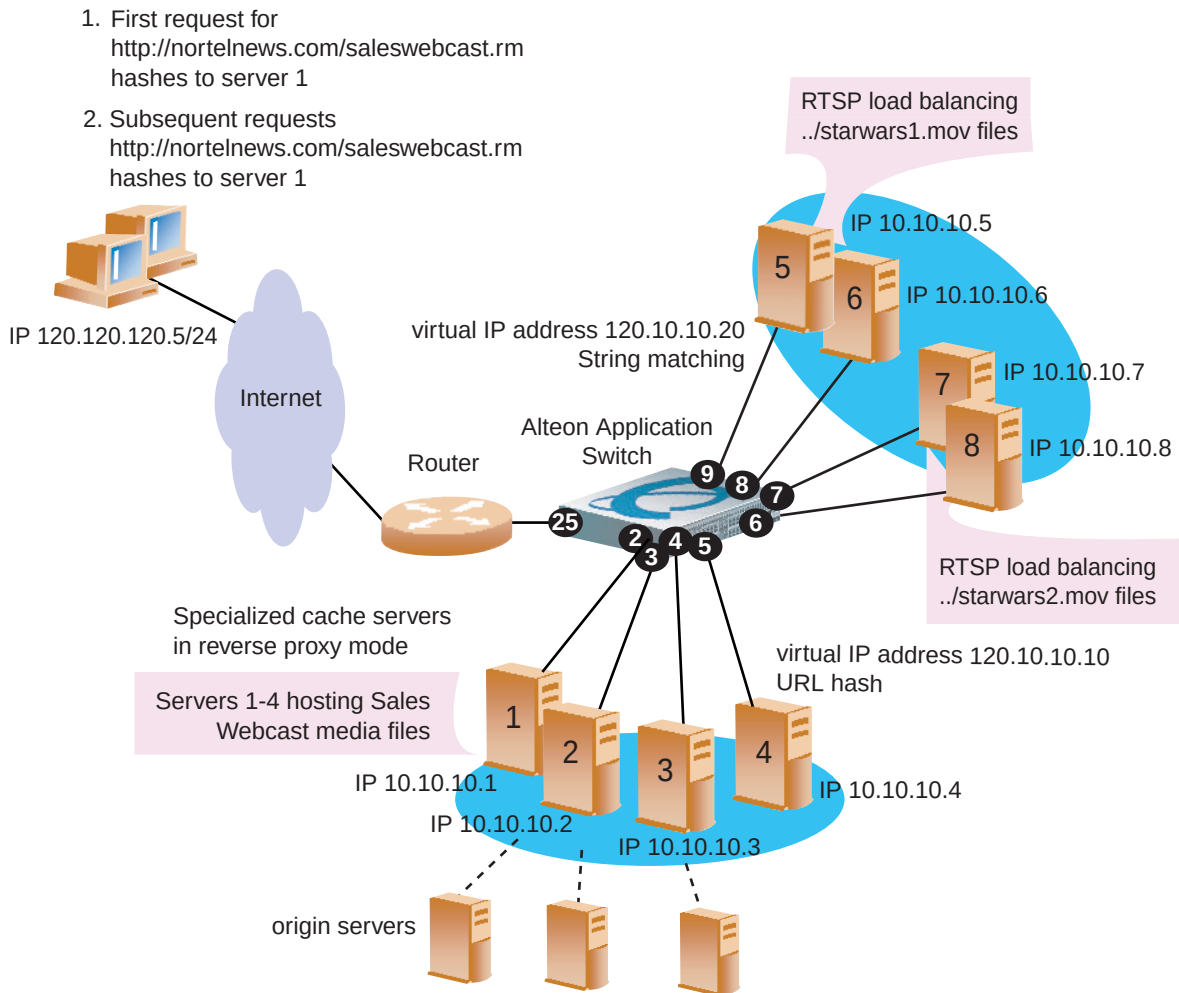


Figure 17-5 Layer 7 RTSP Load Balancing

Follow this procedure to configure the topology illustrated in [Figure 17-5](#):

1. **Before you start configuring RTSP load balancing, configure the application switch for standard server load balancing as described in “Configuring Server Load Balancing” on page 144:**
 - Connect each Media server to the application switch
 - Configure the IP addresses on all devices connected to the switch
 - Configure the IP interfaces on the switch

- Enable server load balancing (/cfg/slb/on)
- Enable client processing at the client port (/cfg/slb/port 1/client ena)
- Enable server processing at the server ports 2 and 7
(for example, /cfg/slb/port 2/server ena)
- Enable Virtual Matrix Architecture (VMA)
- Enable Direct Access Mode (DAM)
- Disable Bandwidth Management
- Disable proxy IP addressing

2. Configure IP addresses for the real servers.

```
>> # /cfg/slb/real 1/rip 10.10.10.1/ena (Define IP address for Real server 1)
>> # /cfg/slb/real 2/rip 10.10.10.2/ena (Define IP address for Real server 2)
>> # /cfg/slb/real 3/rip 10.10.10.3/ena (Define IP address for Real server 3)
>> # /cfg/slb/real 4/rip 10.10.10.4/ena (Define IP address for Real server 4)
>> # /cfg/slb/real 5/rip 10.10.10.5/ena (Define IP address for Real server 5)
>> # /cfg/slb/real 6/rip 10.10.10.6/ena (Define IP address for Real server 6)
>> # /cfg/slb/real 7/rip 10.10.10.7/ena (Define IP address for Real server 7)
>> # /cfg/slb/real 8/rip 10.10.10.8/ena (Define IP address for Real server 8)
```

3. Create a group to support RealNetworks servers.

```
>> # /cfg/slb/group 100 (Define a group)
>>Real Server Group 100# add 1 (Add real server 1)
>>Real Server Group 100# add 2 (Add real server 2)
>>Real Server Group 100# add 3 (Add real server 3)
>>Real Server Group 100# add 4 (Add real server 4)
```

4. Create a group to support QuickTime servers.

```
>> # /cfg/slb/group 200 (Define a group)
>>Real Server Group 200# add 5 (Add real server 5)
>>Real Server Group 200# add 6 (Add real server 6)
>>Real Server Group 200# add 7 (Add real server 7)
>>Real Server Group 100# add 8 (Add real server 8)
```

5. Create a virtual server for group 1 media servers.

Configure a virtual server and select **rtsp** or port **554** as a service for the virtual server.

```
>> # /cfg/slb/virt 1 (Select the virtual server)
>>Virtual Server 1# vip 120.10.10.10 (Set IP address for the virtual server)
>>Virtual Server 1# service 554 (Add the RTSP service for the virtual server)
>>Virtual Server 1 rtsp Service# group 100 (Set the real server group)
>>Virtual Server 1 rtsp Service# ../ena (Enable virtual server)
```

6. Configure URL hash-based RTSP load balancing for group 1 servers.

URL hashing maintains persistency by enabling the client to hash to the same media server.

```
>> Virtual Server 1 rtsp Service# rtspslb hash
```

7. Create another virtual server for group 2 media servers.

Configure a virtual server and select **rtsp** or port **554** as a service for the virtual server.

```
>> # /cfg/slb/virt 2 (Select the virtual server)
>>Virtual Server 2# vip 120.10.10.20 (Set IP address for the virtual server)
>>Virtual Server 2# service 554 (Add the RTSP service for the virtual server)
>>Virtual Server 2 rtsp Service# group 200 (Set the real server group)
>>Virtual Server 2 rtsp Service# ../ena (Enable virtual server)
```

8. Configure string matching-based RTSP load balancing for group 2 servers.

- Enable layer 7 pattern matching.

```
>> Virtual Server 2 rtsp Service# rtspslb pattern
(Enable Layer 7 pattern matching)
```

- Add URL strings.

```
>> # /cfg/slb/layer7/slb/add starwars1.mov (Add URL strings)
>> Server Loadbalance Resource# add starwars2.mov
```

- Apply and save the configuration.

```
>> Server Loadbalance Resource# apply
>> Server Loadbalance Resource# save
```


- Identify the defined string IDs.

```
>> Server Loadbalance Resource# cur
```

For easy configuration and identification, each defined string has an ID attached, as shown in the following example:

Number of entries: three

ID	SLB String
1	any, cont 1024
2	starwars1.mov, cont 1024
3	starwars2.mov, cont 1024

- Add the defined string IDs to the real servers as shown in [Figure 17-5](#).

```
>> # /cfg/slb/real 5/layer7
>> Real server 5 Layer 7 Commands# addlb 2
>> Real server 5# .././real 6/layer7
>> Real server 6 Layer 7 Commands# addlb 2
>> Real server 6# .././real 7/layer7
>> Real server 7 Layer 7 Commands# addlb 3
>> Real server 7# .././real 8/layer7
>> Real server 8 Layer 7 Commands# addlb 3
```

9. Apply and save your configuration.

```
>> Real server 8# apply
>> Real server 8# save
```

Clients retrieving RTSP://nortelnews.com/saleswebcast.rm hash to the same media server—1, 2, 3, or 4.

A client request of the form RTSP://120.10.10.20/..../starwars1.mov is load balanced between RTSP servers 5 and 6 using string matching. A client request of the form RTSP://120.10.10.20/..../starwars2.mov is load balanced between RTSP servers 7 and 8.

Content Intelligent Cache Redirection

Alteon OS allows you to redirect cache requests based on different HTTP header information, such as “Host:” header or “User-Agent” for browser-smart load balancing. For more information on layer 4 cache redirection, see [Chapter 11, “Application Redirection.”](#)

The No Cache/Cache Control for cache redirection feature in Alteon OS allows you to off load the processing of non-cacheable content from cache servers by sending only appropriate requests to the cache server farm. When a Cache-Control header is present in a HTTP 1.1 request, it indicates a client's special request with respect to caching, such as to guarantee up-to-date data from the origin server. If this feature (Cache-Control: no cache directive) is enabled, HTTP 1.1 GET requests are forwarded directly to the origin servers.

NOTE – The term *origin server* refers to the server originally specified in the request.

The HTTP 1.0 Pragma: no-cache header is equivalent to the HTTP 1.1 Cache-Control header. By enabling the Pragma: no-cache header, requests are forwarded to the origin server.

NOTE – For cache redirection, at any given time one HTTP header is supported globally for the entire switch.

This section discusses the following types of cache redirection:

- “URL-Based Cache Redirection” on page 443
- “HTTP Header-Based Cache Redirection” on page 451
- “Browser-Based Cache Redirection” on page 452
- “URL Hashing for Cache Redirection” on page 453
- “RTSP Streaming Cache Redirection” on page 456

URL-Based Cache Redirection

URL parsing for cache redirection operates in a manner similar to URL-based server load balancing except that in cache redirection a virtual server on the switch is the target of all IP/HTTP requests. For information on URL-based server load balancing, see [“URL-Based Server Load Balancing” on page 419](#).

By separating static and dynamic content requests via URL parsing, Alteon OS enables you to send requests with specific URLs or URL strings to designated cache servers. The URL-based cache redirection option allows you to off load overhead processing from the cache servers by only sending appropriate requests to the cache server farm.

NOTE – Both HTTP 1.0 and HTTP 1.1 requests are supported.

Each request is examined and handled as described below:

- If the request is a non-GET request such as HEAD, POST, PUT, or HTTP with cookies, it is not sent to the cache.
- If the request is an ASP or CGI request or a dynamically generated page, it is not sent to the cache.
- If the request contains a Cookie, it can optionally bypass the cache.

Examples of matching string expressions are:

- /product
Any URL that starts with “/product,” including any information in the “/product” directory
- product
Any URL that has the string “product”

Some of the common noncacheable items that you can configure the switch to add to, delete, or modify are:

- Dynamic content files:
 - Common gateway interface files (.cgi)
 - cold fusion files (.cfm), ASP files (.asp)
 - BIN directory
 - CGI-BIN directory
 - SHTML (scripted html)
 - Microsoft HTML extension files (.htx)
 - executable files (.exe)
- Dynamic URL parameters: +, !, %, =, &

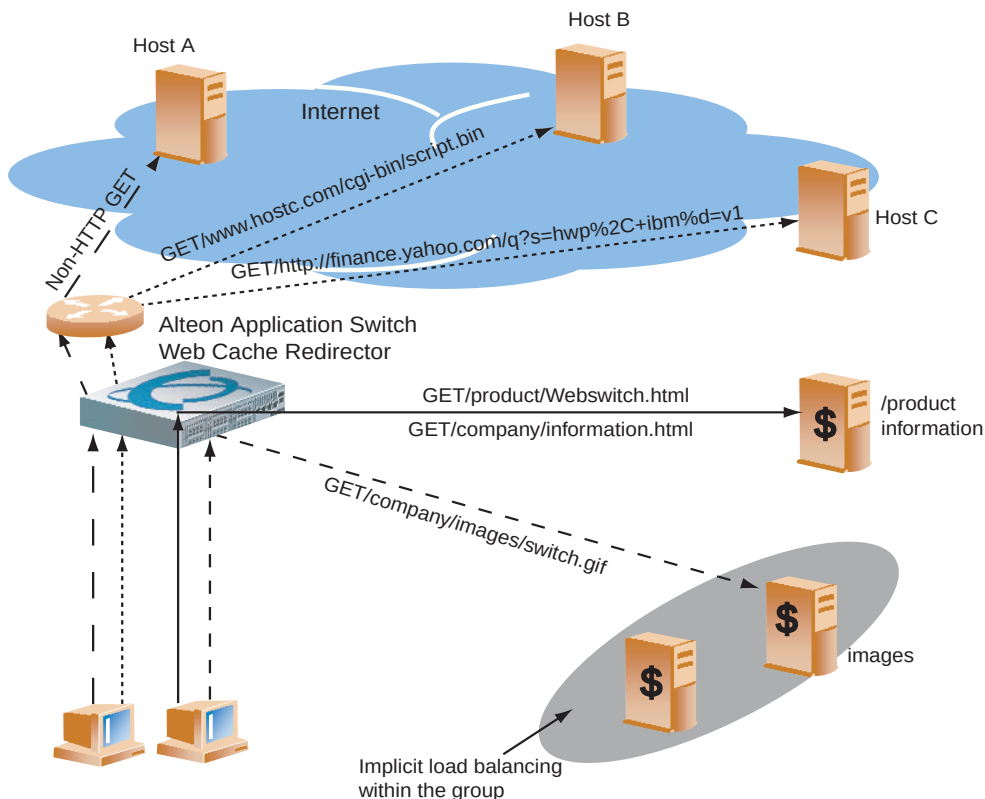


Figure 17-6 URL-Based Cache Redirection

Requests matching the URL are load balanced among the multiple servers, depending on the metric specified for the real server group (least conns is the default).

Network Address Translation Options

URL-based cache redirection supports three types of Network Address Translation (NAT): No NAT, Half NAT, and Full NAT.

■ No NAT

In this NAT method, the traffic is redirected to the cache with the destination MAC address of the virtual server replaced by the MAC address of the cache. The destination IP address remains unchanged, and no modifications are made to the IP address or the MAC address of the source or origin server. This works well for transparent cache servers, which process traffic destined to their MAC address but use the IP address of some other device.

■ Half NAT

In this most commonly used NAT method, the destination IP address is replaced by the IP address of the cache, and the destination MAC address is replaced by the MAC address of the cache. Both the IP address and the MAC address of the source remain unchanged.

■ Full NAT

In this NAT method, the source IP address and the source MAC address are replaced by the IP address and MAC address of the cache. This method works well for proxy cache servers.

Configuring URL-Based Cache Redirection

To configure URL-based cache redirection, perform the following steps:

1. **Before you can configure URL-based cache redirection, configure the switch for basic Server Load Balancing (SLB) with the following tasks:**

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.

For information on how to configure your network for SLB, see [“Server Load Balancing” on page 137](#).

2. **Configure the switch to support basic cache redirection.**

For information on cache redirection, refer to [“Application Redirection” on page 241](#).

3. Configure the parameters and file extensions that bypass cache redirection.

a) Add or remove string IDs that should *not* be cacheable.

```
>> # /cfg/slb/filt 1/adv/addstr|remstr <ID>
>> # /cfg/slb/layer7/slb/add|remove <strings>
```

b) Enable/disable ALLOW for non-GETS (such as HEAD, POST, and PUT) to the origin server, as described below.

```
>> # /cfg/slb/layer7/redirect/urlal ena|dis
```

- ☐ Ena: The switch allows all non-GET requests to the origin server.
- ☐ Dis: The switch compares all requests against the expression table to determine whether the request should be redirected to a cache server or the origin server.

c) Enable/disable cache redirection of requests that contain “cookie:” in the HTTP header.

```
>> # /cfg/slb/layer7/redirect/cookie ena|dis
```

- ☐ Ena: The switch redirects all requests that contain “cookie:” in the HTTP header to the origin server.
- ☐ Dis: The switch compares the URL against the expression table to determine whether the request should be redirected to a cache server or the origin server.

d) Enable/disable cache redirection of requests that contain “Cache-control:no cache” in the HTTP 1.1 header or “Pragma:no cache” in the HTTP 1.0 header to the origin server.

```
>> # /cfg/slb/layer7/redirect/nocache ena|dis
```

- ☐ Ena: The switch redirects all requests that contain Cache-control: no cache in the HTTP 1.1 header or Pragma: no cache in the HTTP 1.0 header to the origin server.
- ☐ Dis: The switch compares the URL against the expression table to determine whether the request should be redirected to a cache server or the origin server.

4. Define the string(s) to be used for cache SLB. Refer to the parameters listed below:

```
>> # /cfg/slb/layer7/slb/add|rem <string>
```

- add: Add a string or a path.
- rem: Remove string or a path.

A default string “any” indicates that the particular server can handle all URL or cache requests. Refer to the following examples:

Example 1: String Starting with the Forwardslash (/)

A string that starts with a forwardslash (/), such as “/images,” indicates that the server will process requests that start out with the “/images” string only.

For example, with the “/images” string, the server will handle these requests:

```
/images/product/b.gif
/images/company/a.gif
/images/testing/c.jpg
```

The server will *not* handle these requests:

```
/company/images/b.gif
/product/images/c.gif
/testing/images/a.gif
```

Example 2: String without the Forwardslash (/)

A string that does not start out with a forwardslash (/) indicates that the server will process any requests that contain the defined string. For example, with the “images” string, the server will process these requests:

```
/images/product/b.gif
/images/company/a.gif
/images/testing/c.jpg
/company/images/b.gif
/product/images/c.gif
/testing/images/a.gif
```

Example 3: String with the Forwardslash (/) Only

If a server is configured with the load balance string (/) only, it will only handle requests to the root directory. For example, the server will handle any files in the ROOT directory:

```
/
/index.htm
/default.asp
/index.shtm
```

- 5. Apply and save your configuration changes.
- 6. Identify the defined string IDs.

```
>> # /cfg/slb/layer7/slb/cur
```

For easy configuration and identification, each defined string has an ID attached, as shown in the following example:

ID	SLB String
1	any
2	.gif
3	/sales
4	/xitami
5	/manual
6	.jpg

- 7. Configure the real server(s) to support cache redirection.

NOTE – If you don't add a defined string (or add the defined string “any”), the server will handle any request.

Add the defined string(s) to the real servers:

```
>> # /cfg/slb/real 2/layer7/addlb <ID>
```

where *ID* is the identification number of the defined string.

The server can have multiple defined strings. For example: “/images”, “/sales”, “.gif”

With these defined strings, the server can handle requests that begin with “/images” or “/sales” and any requests that contain “.gif”.

- 8. Define a real server group and add real servers to the group.

The following configuration combines three real servers into a group:

>> # /cfg/slb/group 1	(Select real server group 1)
>> Real server group 1# add 1	(Add real server 1 to group 1)
>> Real server group 1# add 2	(Add real server 2 to group 1)
>> Real server group 1# add 3	(Add real server 3 to group 1)

9. Configure a filter to support basic cache redirection.

The filter must be able to intercept all TCP traffic for the HTTP destination port and must redirect it to the proper port in the real server group:

```
>> # /cfg/slb/filt <filter number>           (Select the menu for Filter #x)
>> Filter <filter number># sip any             (From any source IP addresses)
>> Filter <filter number># dip any             (To any destination IP addresses)
>> Filter <filter number># proto tcp           (For TCP protocol traffic)
>> Filter <filter number># sport any           (From any source port)
>> Filter <filter number># dport http           (To an HTTP destination port)
>> Filter <filter number># action redir        (Set the action for redirection)
>> Filter <filter number># rport http           (Set the redirection port)
>> Filter <filter number># group 1             (Select real server group 1)
>> Filter <filter number># ena                 (Enable the filter)
```

10. Enable URL-based cache redirection on the same filter.

```
>> # /cfg/slb/filt <filter number>/adv/urlp ena
```

11. Select the appropriate NAT option.

The three NAT options are listed below. For more information about each option, see “[Network Address Translation Options](#)” on page 445.

■ No NAT option:

```
>> # /cfg/slb/filter <filter number>/adv/proxy dis
```

■ Half NAT option:

```
>> # /cfg/slb/filter <filter number>/adv/proxy ena
```

■ Full NAT option:

```
>> # /cfg/slb/pip
>> Proxy IP Address# pip1 12.12.12.12         (Configure proxy IP address)
>> # /cfg/slb/filt <filter number>
>> Filter <filter number># rport 3128         (Specify redirection port)
>> Filter <filter number># adv                 (Select the advance menu)
>> Filter <filter number> Advanced# proxy ena (Enable proxy IP address)
```

For more information on proxy IP addresses, see “[Proxy IP Addresses](#)” on page 156.

12. Create a default filter for noncached traffic on the switch.

```

>> # /cfg/slb/filt <filter number>           (Select the default filter)
>> Filter <filter number># sip any             (From any source IP addresses)
>> Filter <filter number># dip any             (To any destination IP addresses)
>> Filter <filter number># proto any           (For any protocol traffic)
>> Filter <filter number># action allow        (Set the action to allow traffic)
>> Filter <filter number># ena                 (Enable the default filter)
>> Filter <filter number># port <port number> (Assign the default filter to a port)

```

NOTE – When the `proto` parameter is not `tcp` or `udp`, then `sport` and `dport` are ignored.

13. Turn on filtering for the port.

```

>> SLB <port number># filt ena

```

14. Add the filters to the client port.

```

>> SLB <port number># add <filter number>

```

15. Enable Direct Access Mode (DAM) on the switch.

```

>> SLB <port number># ../adv
>> Layer 4 Advanced# direct ena

```

16. Enable, apply, and verify the configuration.

```

>> # /cfg/slb           (Select the SLB Menu)
>> # on                 (Turn SLB on)
>> # apply              (Make your changes active)
>> # cur                (View current settings)

```

Viewing Statistics for URL-Based Cache Redirection

To show the number of hits to the cache server or origin server, use this command:

```

>> # /stats/slb/layer7/redir
Total URL based Web cache redirection stats:
Total cache server hits:                73942
Total origin server hits:               2244
Total none-GETs hits:                   53467
Total 'Cookie: ' hits:                  729
Total no-cache hits:                    43

```

HTTP Header-Based Cache Redirection

To configure the switch for cache direction based on the “Host:” header, use the following procedure:

1. Configure basic SLB.

Before you can configure header-based cache redirection, ensure that the switch has already been configured for basic SLB (see “[Server Load Balancing](#)” on page 137) with the following tasks:

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Assign servers to real server groups.
- Define virtual servers and services.

2. Turn on URL parsing for the filter.

```
>> # /cfg/slb/filt 1/adv/urlp ena
```

3. Enable header load balancing for the Host : header.

```
>> # /cfg/slb/layer7/redir/header ena host
```

4. Define the host names.

```
>> # /cfg/slb/layer7/slb/add ".com"
>> Server Load Balance Resource# add ".org"
>> Server Load Balance Resource# add ".net"
```

5. Apply and save your configuration changes.

6. Identify the string ID numbers with this command.

```
>> # /cfg/slb/layer7/slb/cur
```

Each defined string has an associated ID number.

ID	SLB String
1	any
2	.com
3	.org
4	.net

7. Configure the real server(s) to handle the appropriate load balance string(s).

Add the defined string IDs to the real servers:

```
>> # /cfg/slb/real 2/layer7/addlb <ID>
```

where ID is the identification number of the defined string.

NOTE – If you don't add a defined string (or add ID=1), the server will handle any request.

Browser-Based Cache Redirection

Browser-based cache redirection uses the User-agent : header. To configure browser-based cache redirection, perform the following procedure.

1. Before you can configure header-based cache redirection, ensure that the switch is already configured for basic SLB with the following tasks:

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Assign servers to real server groups.
- Define virtual servers and services.

2. Turn on URL parsing for the filter.

```
>> # /cfg/slb/filt 1/adv/urlp enable
```

3. Enable header load balancing for “User-Agent:” header.

```
>> # /cfg/slb/layer7/redir/header ena useragent
```

4. Define the host names.

```
>> # /cfg/slb/layer7/slb/add "Mozilla"
>> Server Load Balance Resource# add "Internet Explorer"
>> Server Load Balance Resource# add "Netscape"
```

5. Apply and save your configuration changes.

6. Identify the string ID numbers with this command.

```
>> # /cfg/slb/layer7/slb/cur
```

Each defined string has an ID number.

Number of entries: four

ID	SLB String
1	any
2	Mozilla
3	Internet Explorer
4	Netscape

7. Add the defined string IDs to configure the real server(s) to handle the appropriate load balance string(s).

```
>> # /cfg/slb/real 2/layer7/addlb <ID>
```

where ID is the identification number of the defined string.

NOTE – If you don't add a defined string (or add the ID 1), the server will handle any request.

URL Hashing for Cache Redirection

By default, hashing algorithms use the source IP address and/or destination IP address (depending on the application area) to determine content location. For example, firewall load balancing uses both source and destination IP addresses, cache redirection uses only the destination IP address, and SLB uses only the source IP address.

Hashing is based on the URL, including the HTTP Host header (if present), up to a maximum of 255 bytes. You can optimize “cache hits” by using the hashing algorithm to redirect client requests going to the same page of an origin server to a specific cache server.

For example the switch could use the string “nortelnetworks.com/products/2424/” for hashing the following request:

```
GET http://products/2424/ HTTP/1.0
HOST:www.nortelnetworks.com
```

To configure the switch for cache redirection based on a hash key, use the following procedure:

1. Configure basic SLB.

Before you can configure header-based cache redirection, ensure that the switch has already been configured for basic SLB (see [“Server Load Balancing” on page 137](#)) with the following tasks:

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Assign servers to real server groups.
- Define virtual servers and services.
- Configure the load-balancing algorithm to `hash` or `minmiss`.

2. Turn on URL parsing for the filter.

```
>> # /cfg/slb/filt 1/adv/urlp ena
```

3. Enable hash to direct a cacheable URL request to a specific cache server.

By default, the host header field is used to calculate the hash key and URL hashing is disabled.

- **hash ena:** Enables hashing based on the URL and the host header if it is present. Specify the length of the URL to hash into the cache server.

```
>> # /cfg/slb/layer7/redir/hash ena
Enter new hash length [1-255]: 24
```

- **hash disable:** Disables hashing based on the URL. Instead, the host header field to calculate the hash key.

If the host header field does not exist in the HTTP header, then the switch uses the source IP address as the hash key.

Example 1: Hashing on the URL

In this example, URL hashing is enabled. If the Host field does not exist, the specified length of the URL is used to hash into the cache server as shown in [Figure 17-7 on page 455](#). If the Host field exists, the specified length of both the Host field and the URL is used to hash into the cache server. The same URL request goes to the same cache server as shown below:

- Client 1 request `http://www.nortelnetworks.com/sales/index.htm` is directed to cache server 1.
- Client 2 request `http://www.nortelnetworks.com/sales/index.htm` is directed to cache server 1.
- Client 3 request `http://www.nortelnetworks.com/sales/index.htm` is directed to cache server 1.

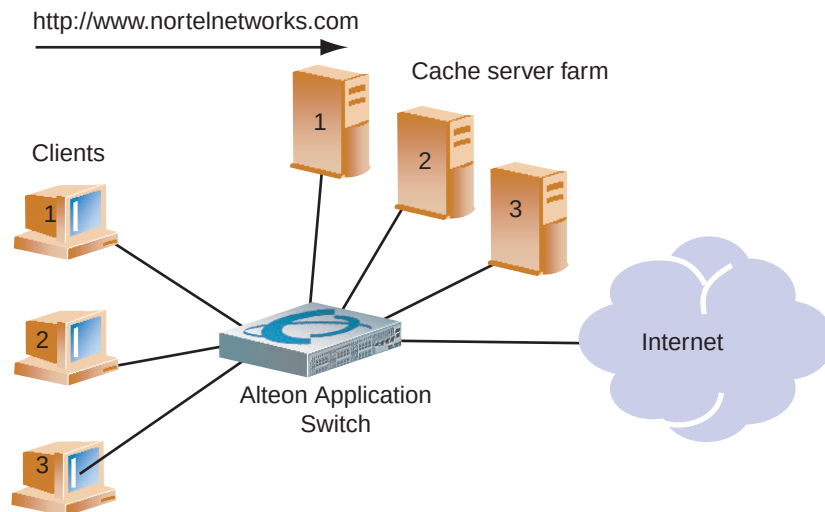


Figure 17-7 URL Hashing for Application Redirection

Example 2: Hashing on the Host Header Field Only

In this example, URL hashing is disabled. If you use the Host header field to calculate the hash key, the same URL request goes to the same cache server:

- Client 1 request `http://www.nortelnetworks.com` is directed to cache server 1.
- Client 2 request `http://www.nortelnetworks.com` is directed to cache server 1.
- Client 3 request `http://www.nortelnetworks.com` is directed to cache server 1.

Example 3: Hashing on the Source IP address

In this example, URL hashing is disabled. Because the host header field does not exist in the HTTP header, the source IP address is used as the hash key and requests from clients 1, 2, and 3 are directed to three different cache servers as shown below.

- Client 1 request `http://www.nortelnetworks.com` is directed to cache server 1.
- Client 2 request `http://www.nortelnetworks.com` is directed to cache server 2.
- Client 3 request `http://www.nortelnetworks.com` is directed to cache server 3.

RTSP Streaming Cache Redirection

RTSP load balancing with the URL hash metric can be used to load balance cache servers that cache multimedia presentations. Since multimedia presentations consume a large amount of Internet bandwidth, and their correct presentation depends upon the real time delivery of the data over the Internet, several caching servers cache the multimedia data.

As a result, the data is available quickly from the cache, when required. The Layer 7 metric of URL hashing directs all requests with the same URL to the same cache server, ensuring that no data is duplicated across the cache servers. All the stream connections and the control connections are switched to the same cache server to facilitate caching of entire presentations.

This section explains Layer 7 support for RTSP Streaming Cache Redirection. For conceptual information on RTSP Streaming Cache Redirection, see [“RTSP Cache Redirection” on page 249](#). For detailed information on two prominent commercial RTSP servers—Real Player and QuickTime—see [“Real Time Streaming Protocol SLB” on page 176](#).

In the scenario illustrated in [Figure 17-8](#), the cache servers are configured for forward proxy mode. The cache servers process the client request even though the destination IP address is not destined for the cache servers.

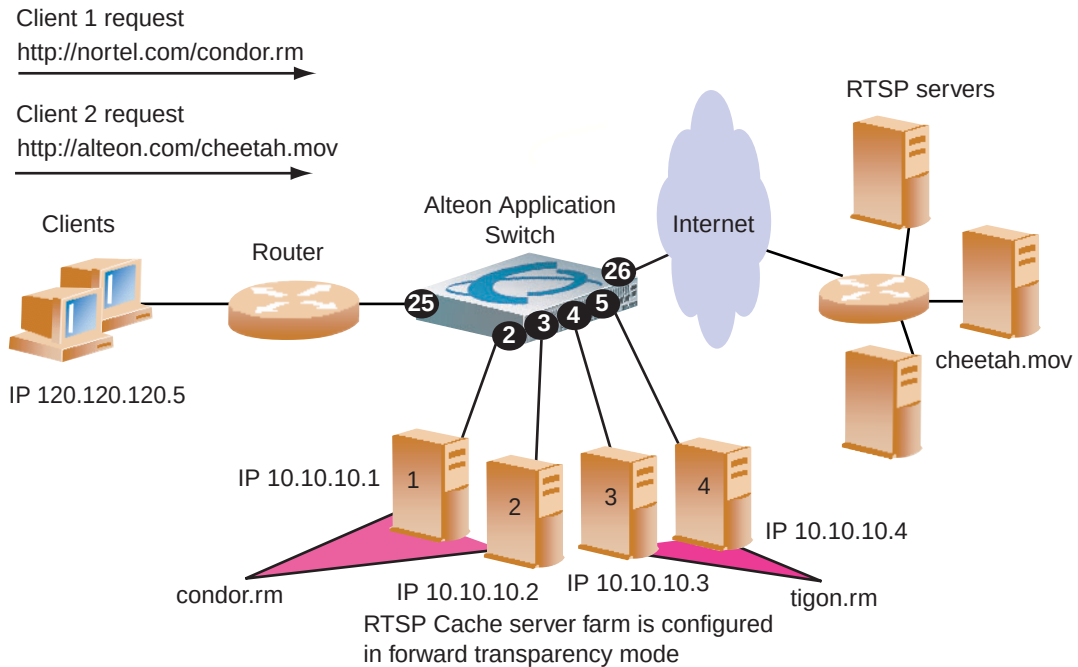


Figure 17-8 Load Balancing RTSP Cache Servers

Follow this procedure to configure to load balance RTSP cache servers for the topology illustrated in [Figure 17-8](#):

1. Before you start configuring the application switch do the following:

- Connect each cache server to the application switch
- Configure the IP addresses on all devices connected to the switch
- Configure the IP interfaces on the switch

2. At the application switch, configure RTSP cache servers and the IP addresses.

```
>> # /cfg/slb/real 1
>> Real server 1# rip 1.1.1.1           (Configure RTSP cache server 1)
>> Real server 1# ena                   (Enable RTSP cache server 1)
>> Real server 1# ../real 2
>> Real server 2# rip 1.1.1.2           (Configure RTSP cache server 2)
>> Real server 2# ena                   (Enable RTSP cache server 2)
>> Real server 2# ../real 3
>> Real server 3# rip 1.1.1.3           (Configure RTSP cache server 3)
>> Real server 3# ena                   (Enable RTSP cache server 3)
>> Real server 3# ../real 4
>> Real server 4# rip 1.1.1.4           (Configure RTSP cache server 4)
>> Real server 4# ena                   (Enable RTSP cache server 4)
```

3. Define a group to load balance the RTSP cache servers.

```
>> # /cfg/slb/group 1
>> Real Server Group 1# add 1           (Add RTSP cache server 1 to group 1)
>> Real Server Group 1# add 2           (Add RTSP cache server 2 to group 1)
>> Real Server Group 1# add 3           (Add RTSP cache server 3 to group 1)
>> Real Server Group 1# add 4           (Add RTSP cache server 4 to group 1)
```

4. Configure a redirection filter.

```
>> # /cfg/slb/filter 100                (Select the menu for filter 100)
>> Filter 100# action redir             (Set the action for redirection)
>> Filter 100# proto tcp                 (Enter TCP protocol)
>> Filter 100# dport rtsp                (Enter service port for RTSP)
>> Filter 100# rport rtsp                (Enter redirection port for RTSP)
>> Filter 100# group 1                   (Select RTSP cache server group 1)
>> Filter 100# adv                       (Select advanced menu for filter 100)
>> Filter 100# Advanced# proxy disable (Disable proxy)
```

5. Enable URL parsing for the redirection filter 100.

```
>> Filter 100 Advanced# urlp ena        (Enable URL parsing)
```

6. Configure a default allow filter to facilitate traffic.

>> # /cfg/slb/filt 2048	(Select a default allow filter 2048)
>> Filter 2048# sip any	(From any source IP addresses)
>> Filter 2048# dip any	(To any destination IP addresses)
>> Filter 2048# ena	(Enable a default allow filter)
>> Filter 2048# action allow	(Set the action to allow normal traffic)

7. Add and enable the redirection filter to the port.

>> # /cfg/slb/port 25	(Select the menu for port 25)
>> SLB Port 25# add 100	(Add RTSP filter 100 to port 25)
>> SLB Port 25# add 2048	(Add default filter 2048 to port 25)
>> SLB Port 25# filt ena	(Enable filtering on port 25)

8. Configure the parameters and file extensions that will bypass RTSP streaming cache redirection. This is for user-defined non-cacheable content.

For example, QuickTime files are non-cacheable—RTSP files with the extension *.mov must bypass the streaming cache redirection. Similarly, you can add other RTSP file extensions (such as *.smil, *.rm, *.ram, and so forth) to bypass the redirection.

>> # /cfg/slb/layer7/redir	(Select the RTSP redirection menu)
>> Web Cache Redirection# add *.mov	(Add non-cacheable RTSP strings)

A client request of the form RTSP:/*.*.mov bypasses the cache servers and instead is routed directly to the original servers.

9. Define the RTSP file extensions to load balance among the cache servers.

>> # /cfg/slb/layer7/slb/add condor.rm
>> Server Load Balance Resource# add tigon.rm

10. Apply and save your configuration changes.

11. Identify the associated ID number for each of the defined RTSP file extension.

>> # /cfg/slb/layer7/slb/cur

ID	SLB String
1	any
2	condor.com
3	tigon.com

12. Assign the URL string ID to the cache servers.

```

>> # /cfg/slb/real 1                               (Select the real server 1)
>> Real Server 1# Layer 7/addlb 2                   (Add the URL string ID 2)
>> Real Server 1 Layer 7 Commands# cfg/slb/real 2
>> Real Server 2# Layer 7/addlb 2                   (Add the URL string ID 2)

>> Real Server 2 Layer 7 Commands# cfg/slb/real 3
>> Real Server 3# Layer 7/addlb 3                   (Add the URL string ID 3)
>> Real Server 3 Layer 7 Commands# cfg/slb/real 4
>> Real Server 4# Layer 7/addlb 3                   (Add the URL string ID 3)

```

13. Apply and save the configuration.

```

>> Real Server 4 Layer 7 Commands# apply
>> Real Server 4 Layer 7 Commands# save

```

Client requests `condor.rm` or `tigon.rm` are retrieved from the local cache servers 1 or 2 and 3 or 4 respectively; however, a client request `cheetah.mov` bypasses the local cache servers and is forwarded to the original server.

Exclusionary String Matching for Real Servers

URL-based SLB and application redirection can match or exclude up to 128 strings. Examples of strings are as follows:

- “/product,” matches URLs that starts with /product.
- “product,” matches URLs that have the string “product” anywhere in the URL.

You can assign one or more strings to each real server. When more than one URL string is assigned to a real server, requests matching any string are redirected to that real server. There is also a special string known as “any” that matches all content.

Alteon OS also supports *exclusionary* string matching. Using this option, you can define a server to accept any requests regardless of the URL, *except* requests with a specific string.

NOTE – Once exclusionary string matching is enabled, clients cannot access the URL strings that are added to that real server. This means you cannot configure a dedicated server to receive a certain string and, at the same time, have it exclude other URL strings. The exclusionary feature is enabled per server, not per string.

For example, the following strings are assigned to a real server:

```
string 1 = cgi
string 2 = NOT cgi/form_A
string 3 = NOT cgi/form_B
```

As a result, all cgi scripts are matched except form_A and form_B.

Configuring for Exclusionary URL String Matching

This configuration example shows you how to configure a server to handle any requests *except* requests that contain the string “test” or requests that start with “/images” or “/product”.

To configure exclusionary URL string matching, perform the following procedure:

1. **Before you can configure URL string matching, ensure that the switch has already been configured for basic SLB:**
 - Assign an IP address to each of the real servers in the server pool.
 - Define an IP interface on the switch.
 - Define each real server.
 - Assign servers to real server groups.
 - Define virtual servers and services.
 - Enable SLB.

For information on how to configure your network for server load balancing, see [Chapter 9, “Server Load Balancing.”](#)

2. **Add the load balancing strings (for example `test`, `/images`, and `/product`) to the real server.**

```
>> # /cfg/slb/layer7/slb/add test
>> Server Loadbalance Resource# add "/images"
>> Server Loadbalance Resource# add "/product"
```

3. **Apply and save the configuration.**

4. **Identify the IDs of the defined strings.**

```
>> Server Loadbalance Resource# cur
```

ID	SLB String
1	any
2	test
3	/images
4	/product

5. **Assign the URL string ID to the real server.**

```
>> Real Server 1 Layer 7 commands# addlb 2
>> Real Server 1 Layer 7 commands# addlb 3
>> Real Server 1 Layer 7 commands# addlb 4
```

6. **Enable the exclusionary string matching option.**

```
>> Real Server 1 Layer 7 commands# exclude enable
```

If you configured a string “any” and enabled the exclusion option, the server will not handle any requests. This has the same effect as disabling the server.

Regular Expression Matching

Regular expressions are used to describe patterns for string matching. They enable you to match the exact string, such as URLs, host names, or IP addresses. It is a powerful and effective way to express complex rules for Layer 7 string matching. Both Layer 7 HTTP SLB and cache redirection can use regular expressions as a resource. Configuring for regular expressions can enhance content-based switching in the following areas:

- HTTP header matching
- URL matching

Standard Regular Expression Characters

The following is a list of standard regular expression special characters that are supported in Alteon OS:

Table 17-1 Standard Regular Expression Special Characters

Construction	Description
*	Matches any string of zero or more characters
.	Matches any single character
+	Matches one or more occurrences of the pattern it follows
?	Matches zero or one occurrences of its followed pattern
\$	Matches the end of a line
\	Escape the following special character
[abc]	Matches any of the single character inside the bracket
[^abc]	Matches any single character except those inside the bracket

Use the following rules to describe patterns for string matching:

- Supports one layer of parenthesis.
- Supports only single “\$” (match at end of line) which must appear at the end of the string. For example, “abc\$*def” is not supported.
- Size of the user input string must be 40 characters or less.

- Size of the regular expression structure after compilation cannot exceed 43 bytes for load balancing strings and 23 bytes for cache redirection. The size of regular expression after compilation varies, based on regular expression characters used in the user input string.
- Use “/” at the beginning of the regular expression. Otherwise a regular expression will have “*” prefixed to it automatically. For example, “html/*\.” appears as “*html/*\.”.
- Incorrectly or ambiguously formatted regular expressions are rejected instantly. For example:
 - where a “+” or “?” follows a special character like the “*”
 - A single “+” or “?” sign
 - Unbalanced brackets and parenthesis

Configuring Regular Expressions

The regular expression feature is applicable to both path strings used for URL-based server load balancing, and expression strings used for URL-based application redirection. Configure regular expressions at the following CLI prompts:

```
/cfg/slb/layer7/slb/add
```

As a result, both HTTP SLB and application redirection can use regular expression as the resource.

NOTE – The more complex the structure of the string, the longer it will take for the server to load balance the incoming packets.

Content Precedence Lookup

The Layer 7 Precedence Lookup feature in Alteon OS allows you to give precedence to one Layer 7 parameter over another and selectively decide which parameter should be analyzed first.

The Content Precedence Lookup feature allows you to combine up to two Layer 7 load balancing mechanisms. You can specify which types of Layer 7 content to examine, the order in which they are examined, and a logical operator (and/or) for their evaluation.

The following Layer 7 content types can be specified:

- URL SLB
- HTTP Host
- Cookie
- Browsers (User agent)
- URL hash
- Header hash

Using the above content types with the *and* and *or* operators, the application switch is configured to refine HTTP-based server load balancing multiple times on a single client HTTP request in order to bind it to an appropriate server. Typically, when you combine two content types with an operator (and/or), URL hash and Header hash are used in combination with Host, Cookie, or Browser content types. For example, the following types of load balancing can be configured using the Content Precedence Lookup feature:

- Virtual host and/or URL-based load balancing
- Cookie persistence and URL-based load balancing
- Cookie load balancing and/or URL-based load balancing
- Cookie persistence and HTTP SLB together in the same service
- Multiple HTTP SLB process type on the same service

NOTE – Cookie persistence can also be combined with the Layer 7 content types. For more information on cookie persistence, see [Chapter 18, “Persistence.”](#)

For example, the Content Precedence Lookup feature can be used in the following scenarios:

- If the client request is sent without a cookie and if no HTTP SLB is configured, then the switch binds the request to the real server using normal SLB.
- If the client request is sent without a cookie, but HTTP SLB is configured on the switch, then the request is bound to real server based on HTTP SLB.
- If the client request is sent with a cookie, and a real server associated to the cookie is found in the local session table, then the request will stay bound to that real server.

Requirements

- Enable Direct Access Mode (DAM), or configure proxy IP address if DAM is disabled.
- Enable delayed binding.

Using the *or* and *and* Operators

Figure 17-9 shows a network with real servers 1 and 3 configured for URL SLB and real servers 2 and 3 configured for HTTP Host SLB.

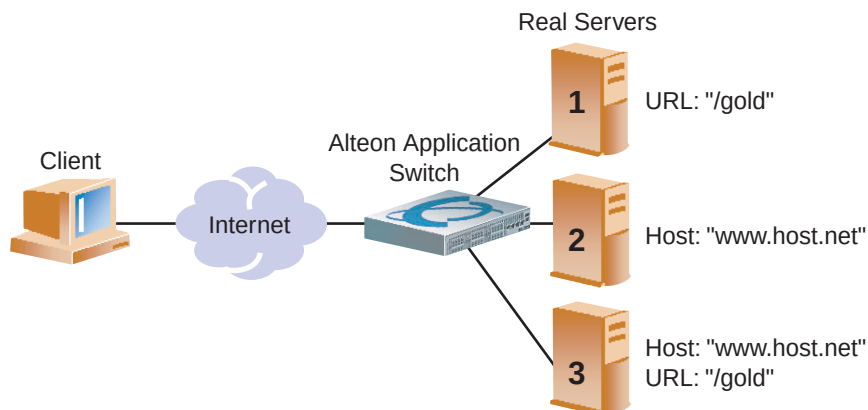


Figure 17-9 Content Precedence Lookup Protectors Example

If the Content Precedence Lookup feature is configured with the *or* and *and* operators, the request from the client is as follows:

- HTTP Host *or* URL SLB

The HTTP Host header takes precedence because it is specified first. If there is no Host Header information and because *or* is the operator, the URL string is examined next.

- If a request from a client contains no Host Header but has a URL string (such as “/gold”), the request is load balanced among Server 1 or Server 3.
- If a request from a client contains a Host Header, then the request is load balanced among Server 2 and Server 3. The URL string is ignored because the HTTP Host was specified and matched first.

■ HTTP Host *and* URL SLB

The HTTP Host header takes precedence because it is specified first. Because *and* is the operator, both a Host Header and URL string are required. If either is not available, the request is dropped.

- If a request from a client contains a URL string (such as “/gold”) but not a Host Header, it is not served by any real server.
- If a request from a client contains a URL string (such as “/gold”) and Host Header, it is served only by real server 3.

Assigning Multiple Strings

Figure 17-10 shows an example of a company providing content for two large customers: Customers A and B. Customer A uses `www.a.com` as their domain name, and Customer B uses `www.b.com`.

The company has a limited number of public IP addresses and wishes to assign them on a very conservative basis. As a result, the company implements virtual hosting by advertising a single virtual server IP address that includes both customers' Web sites. Additionally, the hosting company assigns only one service (HTTP port 80) to support the virtual server.

The virtual hosting company wishes to maintain the flexibility to allow different types of content to be placed on different servers. To make most efficient use of their server resources, they separate their servers into two groups, using their fastest servers to process dynamic content (such as .cgi files) and their slower servers to process all static content (such as .jpg files).

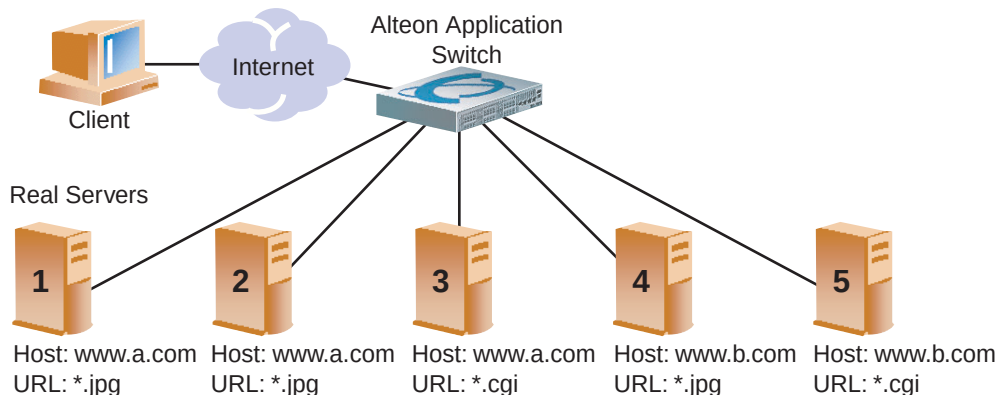


Figure 17-10 Content Precedence Lookup Multiple Strings Example

To configure content precedence lookup for the example in [Figure 17-10](#), the hosting company groups all the real servers into one real server group even though different servers provide services for different customers and different types of content. In this case, the servers are set up for the following purpose:

Table 17-2 Real Server Content

Server	Customer	Content
Server 1	Customer A	Static .jpg files
Server 2	Customer A	Static .jpg files
Server 3	Customer A	Dynamic .cgi files
Server 4	Customer B	Static .jpg files
Server 5	Customer B	Dynamic .cgi files

When a client request is received with `www.a.com` in the Host Header and `.jpg` in the URL, the request will be load balanced between Server 1 and Server 2.

To accomplish this configuration, you must assign multiple strings (a Host Header string and a URL string) for each real server.

Layer 7 Deny Filters

Alteon OS allows you to secure your switch from virus attacks by configuring the switch with a list of potential offending string patterns (HTTP URL request). The switch examines the HTTP content of the incoming client request for the matching string pattern. If the matching virus pattern is found, then the packet is dropped and a reset frame is sent to the offending client. SYSLOG messages and SNMP traps are generated warning operators of a possible attack.

A layer 7 deny filter is basically a deny filter except that the deny action is delayed until HTTP content is examined to see if the packet should be denied.

Figure 17-11 shows an incoming client request with a virus string. The application switch is configured for Layer 7 deny filter, so it blocks the incoming packet with the virus string and prevents it from entering the network.

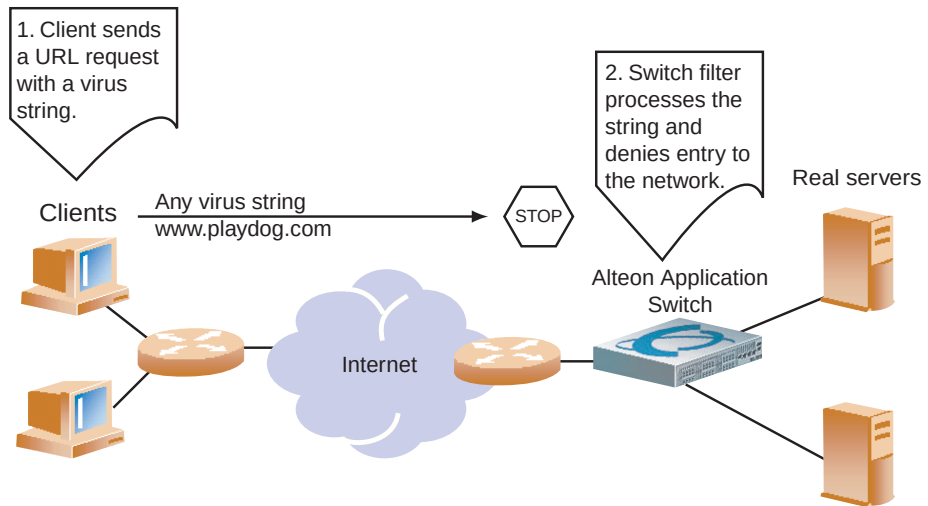


Figure 17-11 Configuring Layer 7 Deny Filter

Configuring a Layer 7 Deny Filter

1. **Before you can configure Layer 7 deny filter, ensure that the switch has already been configured for basic switch functions:**

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.

For information on how to configure your network for the above tasks, see [Chapter 9, “Server Load Balancing.”](#)

2. **Define the virus string patterns or offending HTTP URL request to be blocked.**

```
>> # /cfg/slb/layer7/slb/add ida (Define the code red virus string)
>> Server Loadbalance resource# add %c1%9c (Define the code blue virus string)
>> Server Loadbalance resource# add %c0%af (Define the code blue virus string)
>> Server Loadbalance resource# add playdog.com (Define the offending URL request)
```

3. **Apply and save the configuration.**

4. Identify the IDs of the defined strings.

```
>> Server Loadbalance resource# cur
```

Number of entries: four

ID	SLB String
1	ida
2	%c1%9c
3	%c0%af
4	playdog.com

5. Assign the URL string ID from [Step 4](#) to the filter.

```
>> Filter 1 Advanced L7deny# addstr 1      (Add the code red virus string)
>> Filter 1 Advanced L7deny# addstr 2      (Add the code blue virus string)
>> Filter 1 Advanced L7deny# addstr 3      (Add the code blue virus string)
>> Filter 1 Advanced L7deny# addstr 4      (Add the offending URL request)
```

6. Select the filter and enable the filter action to deny.

```
>> # /cfg/slb/filt 1                      (Select the filter)
>> Filter 1 # action deny                  (Set the filter action to deny)
```

7. Enable the Layer 7 deny option.

```
>> Filter 1 Advanced# l7deny ena          (Enable the Layer 7 deny filter)
```

8. Apply and save the configuration.**9. Apply the filter to the client port.**

If the incoming client requests are on port 3, then add the filter to the port.

```
>> # /cfg/slb/port 3                      (Select the client port)
>> SLB Port 3# filt ena                    (Enable filtering on the port)
>> SLB Port 3# add 1                       (Add the Layer 7 filter to the port)
```



CHAPTER 18

Persistence

The Alteon OS persistence feature ensures that all connections from a specific client session reach the same real server, even when Server Load Balancing (SLB) is used.

The following topics are addressed in this chapter:

- [“Overview of Persistence” on page 472](#). This section gives an overview of persistence and the different types of persistence methods implemented in Alteon OS.
- [“Cookie-Based Persistence” on page 474](#). The use of cookie persistence provides a mechanism for inserting a unique key for each client of a virtual server. This feature is only used in nonsecure socket layer (non-SSL) connections. This section discusses in detail how persistence is maintained between a client and a real server using different types of cookies.
- [“Server-Side Multi-Response Cookie Search” on page 486](#). This section explains how to configure the switch to look through multiple HTTP responses from the server to achieve cookie-based persistence.
- [“SSL Session ID-Based Persistence” on page 487](#). This section explains how an application server and client communicate over an encrypted HTTP session.

Overview of Persistence

In a typical SLB environment, traffic comes from various client networks across the Internet to the virtual server IP address on the Alteon Application Switch. The switch then load balances this traffic among the available real servers.

In any authenticated Web-based application, it is necessary to provide a persistent connection between a client and the content server to which it is connected. Because HTTP does not carry any state information for these applications, it is important for the browser to be mapped to the same real server for each HTTP request until the transaction is complete. This ensures that the client traffic is not load balanced mid-session to a different real server, forcing the user to restart the entire transaction.

Persistence-based SLB enables the network administrator to configure the network to redirect requests from a client to the same real server that initially handled the request. Persistence is an important consideration for administrators of e-commerce Web sites, where a server may have data associated with a specific user that is not dynamically shared with other servers at the site.

In Alteon OS, persistence can be based on the following characteristics: source IP address, cookies, and Secure Sockets Layer (SSL) session ID.

Using Source IP Address

Until recently, the only way to achieve TCP/IP session persistence was to use the source IP address as the key identifier. There are two major conditions which cause problems when session persistence is based on a packet's IP source address:

- **Many clients sharing the same source IP address (proxied clients):** Proxied clients appear to the switch as a single source IP address and do not take advantage of SLB on the switch. When many individual clients behind a firewall use the same proxied source IP address, requests are directed to the same server, without the benefit of load balancing the traffic across multiple servers. Persistence is supported without the capability of effectively distributing traffic load.

Also, persistence is broken if you have multiple proxy servers behind the application switch performing SLB. The application switch changes the client's address to different proxy addresses as attempts are made to load balance client requests.

- **Single client sharing a pool of source IP addresses:** When individual clients share a pool of source IP addresses, persistence for any given request cannot be assured. Although each source IP address is directed to a specific server, the source IP address itself is randomly selected, thereby making it impossible to predict which server will receive the request. SLB is supported, but without persistence for any given client.

Using Cookies

Cookies are strings passed via HTTP from servers to browsers. Based on the mode of operation, cookies are inserted by either the application switch or the server. After a client receives a cookie, a server can poll that cookie with a GET command, which allows the querying server to positively identify the client as the one that received the cookie earlier.

The cookie-based persistence feature solves the proxy server problem and gives better load distribution at the server site. In the application switch, cookies are used to route client traffic back to the same physical server to maintain session persistence.

Using SSL Session ID

The SSL session ID is effective only when the server is running SSL transactions. Because of the heavy processing load required to maintain SSL connections most network configurations use SSL only when it is necessary. Persistence based on SSL Session ID ensures completion of complex transactions in proxy server environments. However, this type of persistence does not scale on servers because of their computational requirements.

Cookie-Based Persistence

Cookies are a mechanism for maintaining state between clients and servers. When the server receives a client request, the server issues a *cookie*, or token, to the client, which the client then sends to the server on all subsequent requests. Using cookies, the server does not require authentication, the client IP address, or any other time-consuming mechanism to determine that the user is the same user that sent the original request.

In the simplest case, the cookie may be just a “customer ID” assigned to the user. It may be a token of trust, allowing the user to skip authentication while his or her cookie is valid. It may also be a key that associates the user with additional state data that is kept on the server, such as a shopping cart and its contents. In a more complex application, the cookie may be encoded so that it actually contains more data than just a single key or an identification number. The cookie may contain the user’s preferences for a site that allows their pages to be customized.

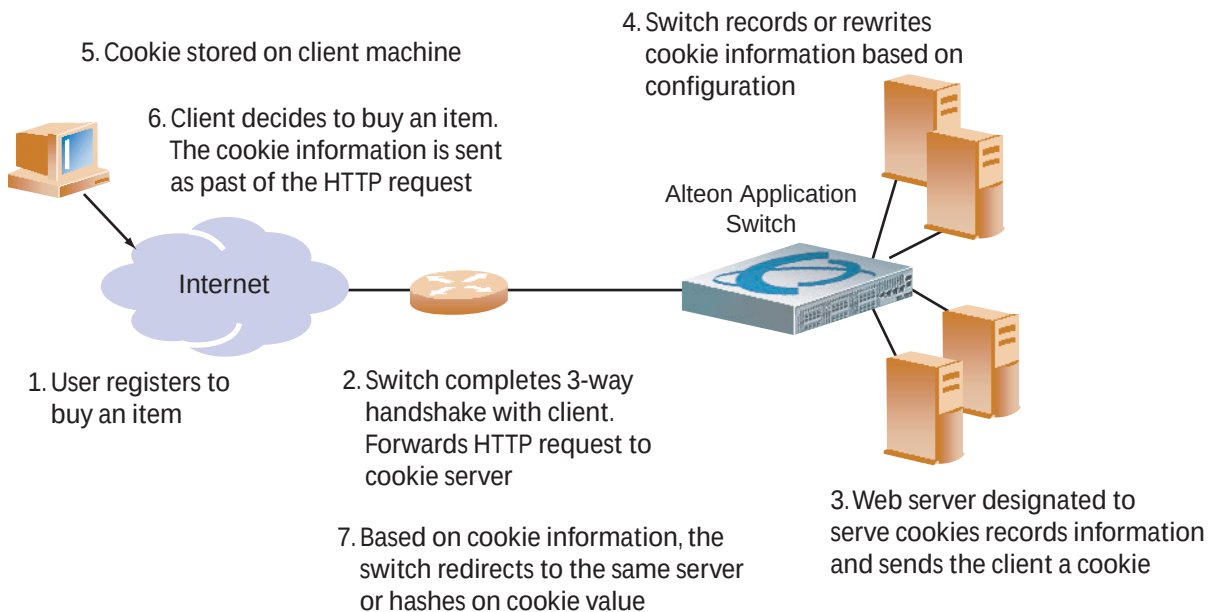


Figure 18-1 Cookie-Based Persistence: How It Works

The following topics discussing cookie-based persistence are detailed in this section:

- “Permanent and Temporary Cookies” on page 475
- “Cookie Formats” on page 475
- “Cookie Properties” on page 476
- “Client Browsers that Do Not Accept Cookies” on page 476
- “Cookie Modes of Operation” on page 477
- “Configuring Cookie-Based Persistence” on page 480

Permanent and Temporary Cookies

Cookies can either be permanent or temporary. A *permanent cookie* is stored on the client's browser, as part of the response from a Web site's server. It will be sent by the browser when the client makes subsequent requests to the same site, even after the browser has been shut down. A *temporary cookie* is only valid for the current browser session. Similar to a SSL Session-based ID, the temporary cookie expires when you shut down the browser. Based on RFC 2109, any cookie without an expiration date is a temporary cookie.

Cookie Formats

A cookie can be defined in the HTTP header (the recommended method) or placed in the URL for hashing. The cookie is defined as a “Name=Value” pair and can appear along with other parameters and cookies. For example, the cookie “SessionID=1234” can be represented in one of the following ways:

- In the HTTP Header

```
Cookie: SesssionID=1234
Cookie: ASP_SESSIONID=POIUHKJHLKHD
Cookie: name=john_smith
```

The second cookie represents an Active Server Page (ASP) session ID. The third cookie represents an application-specific cookie that records the name of the client.

- Within the URL

```
http://www.mysite.com/reservations/SessionID=1234
```

Cookie Properties

Cookies are configured on the application switch by defining the following properties:

- Cookie names of up to 20 bytes
- The offset of the cookie value within the cookie string
For security, the real cookie value can be embedded somewhere within a longer string. The offset directs the application switch to the starting point of the real cookie value within the longer cookie string.
- Length of the cookie value
This defines the number of bytes to extract for the cookie value within a longer cookie string.
- Whether to find the cookie value in the HTTP header (the default) or the URL
- Cookie values of up to 64 bytes for hashing
Hashing on cookie values is used only with the passive cookie mode ([“Passive Cookie Mode” on page 478](#)), using a temporary cookie. The switch mathematically calculates the cookie value using a hash algorithm to determine which real server should receive the request.
- An asterisk (*) in cookie names for wildcards
For example, Cookie name = ASPsession*

Client Browsers that Do Not Accept Cookies

Under normal conditions, most browsers are configured to accept cookies. However, if a client browser is not configured to accept cookies, you must use hash as the load-balancing metric to maintain session persistence.

With cookie-based persistence enabled, session persistence for browsers that do not accept cookies will be based on the source IP address. However, individual client requests coming from a proxy firewall will appear to be coming from the same source IP address. Therefore, the requests will be directed to a single server, resulting in traffic being concentrated on a single real server instead of load balanced across the available real servers.

Cookie Modes of Operation

Alteon OS supports the following modes of operation for cookie-based session persistence: *insert*, *passive*, and *rewrite* mode. The following table shows the differences among the modes:

Table 18-1 Comparison Among the Three Cookie Modes

Cookie Mode	Configuration Required	Cookie Location	Uses Switch Session Entry
Insert Cookie	Switch only	HTTP Header	No
Passive Cookie	Server and Switch	HTTP Header or URL	Yes
Rewrite Cookie	Server and Switch	HTTP Header	No

Each of the modes are explained in detail in the following sections.

Insert Cookie Mode

In the insert cookie mode, the application switch generates the cookie value on behalf of the server. Because no cookies are configured at the server, the need to install cookie server software on each real server is eliminated.

In this mode, the client sends a request to visit the Web site. The application switch performs load balancing and selects a real server. The real server responds without a cookie. The application switch inserts a cookie and forwards the new request with the cookie to the client.

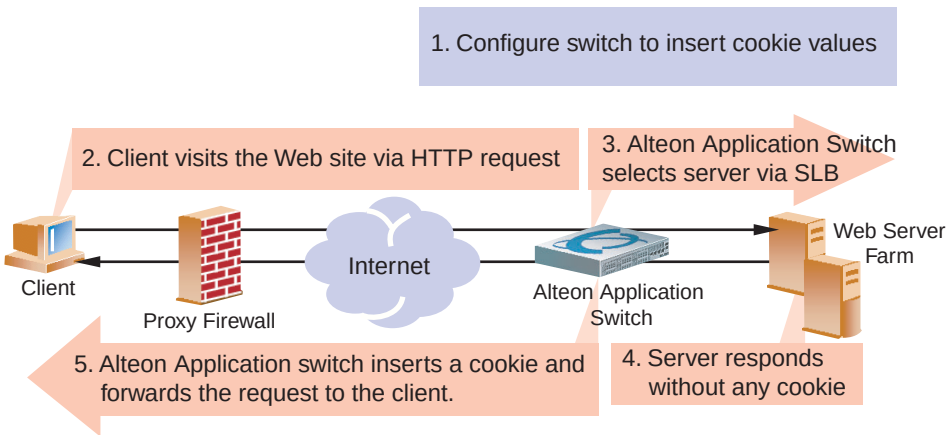


Figure 18-2 Insert Cookie Mode

Passive Cookie Mode

In Passive Cookie mode, when the client first makes a request, the switch selects the server based on the load-balancing metric. The real server embeds a cookie in its response to the client. The switch records the cookie value and matches it in subsequent requests from the same client.

NOTE – The passive cookie mode is recommended for temporary cookies. However, you can use this mode for permanent cookies if the server is embedding an IP address.

The following figure shows passive cookie mode operation:

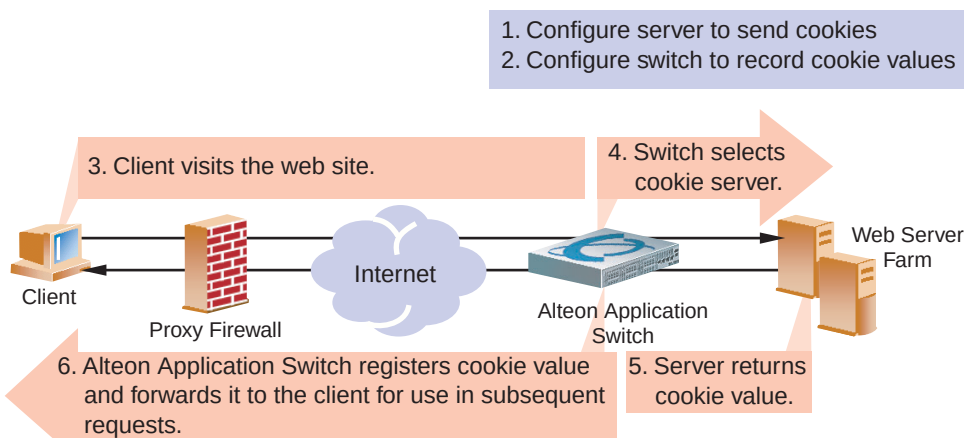


Figure 18-3 Passive Cookie Mode

Subsequent requests from Client 1 with the same cookie value will be sent to the same real server (RIP 1 in this example).

Rewrite Cookie Mode

In rewrite cookie mode, the application switch generates the cookie value on behalf of the server, eliminating the need for the server to generate cookies for each client.

Instead, the server is configured to return a special persistence cookie which the switch is configured to recognize. The switch then intercepts this persistence cookie and rewrites the value to include server-specific information before sending it on to the client. Subsequent requests from the same client with the same cookie value are sent to the same real server.

Rewrite cookie mode requires at least eight bytes in the cookie header. An additional eight bytes must be reserved if you are using cookie-based persistence with Global Server Load Balancing (GSLB).

NOTE – Rewrite cookie mode only works for cookies defined in the HTTP header, not cookies defined in the URL.

Example: The following figure shows rewrite cookie mode operation:

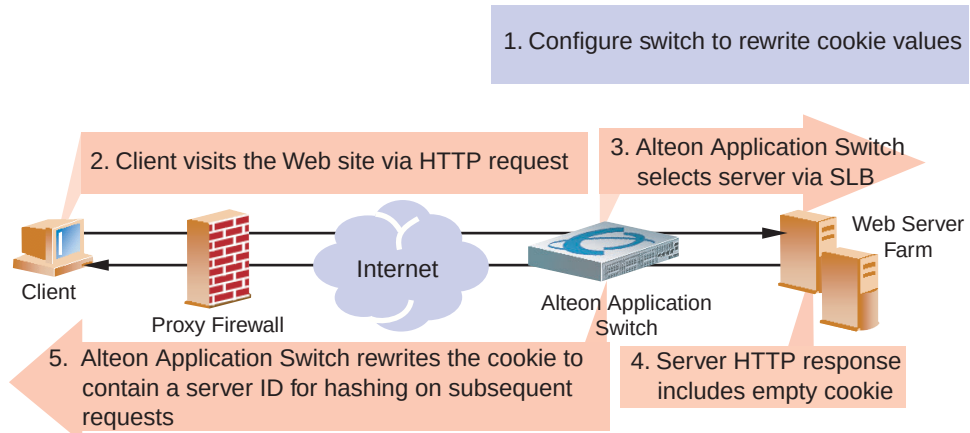


Figure 18-4 Rewrite Cookie Mode

NOTE – When the application switch rewrites the value of the cookie, the rewritten value represents the responding server; that is, the value can be used for hashing into a real server ID or it can be the real server IP address. The rewritten cookie value is encoded.

Configuring Cookie-Based Persistence

1. **Before you can configure cookie-based persistence, you need to configure the switch for basic SLB. This includes the following tasks:**

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Configure each real server with its IP address, name, weight, and so forth.
- Assign servers to real server groups.
- Define virtual servers and services.

For information see [“Server Load Balancing” on page 137](#).

2. **Either enable Direct Access Mode (DAM), or disable DAM and specify proxy IP address(es) on the client port(s).**

- Enable DAM for the switch.

<pre>>> # /cfg/slb/adv/direct ena</pre>	<i>(Enable Direct Access Mode on switch)</i>

- Disable DAM and specify proxy IP address(es) on the client port(s).

<pre>>> # /cfg/slb/adv/direct disable</pre>	<i>(Disable DAM on the switch)</i>
<pre>>> # /cfg/slb/port 1</pre>	<i>(Select network port 1)</i>
<pre>>> # pip 200.200.200.68</pre>	<i>(Set proxy IP address for port 1)</i>

NOTE – If Virtual Matrix Architecture (VMA) is enabled on the switch, you must configure a unique proxy IP address for every port (except port 9).

3. **If proxy IP addresses are used, make sure server processing is disabled on the server port.**

<pre>>> # /cfg/slb/port 1</pre>	<i>(Select switch port 1)</i>
<pre>>> # server dis</pre>	<i>(Disable server processing on port 1)</i>

4. Select the appropriate load-balancing metric for the real server group.

```
>> # /cfg/slb/group 2/metric hash (Select hash as server group metric)
```

- If embedding an IP address in the cookie, select `roundrobin` or `leastconns` as the metric.
- If you are *not* embedding the IP address in the cookie, select `hash` as the metric in conjunction with a cookie assignment server.

While you may experience traffic concentration using the `hash` metric with a cookie assignment server, using a `hash` metric without a cookie assignment server will cause traffic concentration on your real servers.

5. Enable cookie-based persistence on the virtual server service.

In this example, cookie-based persistence is enabled for service 80 (HTTP).

```
>> # /cfg/slb/virt 1/service 80/pbind
Current persistent binding mode: disabled
Enter cookie|sslid|disable persistence mode: cookie
```

Once you specify `cookie` as the mode of persistence, you will be prompted for the following parameters:

```
Enter insert|passive|rewrite cookie persistence mode [i/p/r]: p
Enter cookie name: CookieSession1
Enter starting point of cookie value [1-64]: 1
Enter number of bytes to extract [0-64]: 8
Look for cookie in URI [e|d]: d
Set multiple response count [1-16]: 1
```

- Cookie-based persistence mode: `insert`, `passive` or `rewrite`
- Cookie name
- Starting point of the cookie value
- Number of bytes to be extracted
- Look for cookie in the URI [`e` | `d`]

If you want to look for cookie name/value pair in the URI, enter `e` to enable this option. To look for the cookie in the HTTP header, enter `d` to disable this option.

■ Set multiple response count

This parameter is set for passive mode only. Typically, the application switch searches the first HTTP response packet from the server and, if a persistence cookie is found, sets up a persistent connection between the server and the client. While this approach works for most servers, some customers with complex server configurations might send the persistence cookie a few responses later. In order to achieve cookie-based persistence in such cases, Alteon OS allows the network administrator to configure the switch to look through multiple HTTP responses from the server. The switch looks for the persistence cookie in the specified number of responses (each of them can be multi-frame) from the server.

Setting Expiration Timer for Insert Cookie

If you configure for insert cookie persistence mode, then you will be prompted for cookie expiration timer. The expiration timer specifies a date string that defines the valid life time of that cookie. The expiration timer for insert cookie can be of the following types:

■ Absolute timer

The syntax for the absolute timer is `MM/dd/yy[@hh:mm]`. The date and time is based on RFC 822, RFC 850, RFC 1036, and RFC 1123, with the variations that the only legal time zone is GMT. Once the expiration date is met, the cookie is not stored or given out. For example,

```
Enter cookie expiration: 12/31/01@11:59
Current persistent binding for http: disabled
New persistent binding for http: cookie
New cookie persistence mode: insert
Inserted cookie expires on Mon 12/31/01 at 11:59
```

■ Relative timer

This timer defines the elapsed time from when the cookie was created. The syntax for the relative timer is `days[:hours[:minutes]]`. For example,

```
Enter cookie expiration: 32:25:61
Current persistent binding for http: disabled
New persistent binding for http: cookie
New cookie persistence mode: insert
Inserted cookie expires after 33 days 2 hours 1 minutes
```

NOTE – If the cookie expiration timer is not specified, the cookie will expire when the user's session ends.

Example 1: Setting the Cookie Location

In this example, the client request has two different cookies labeled “UID.” One exists in the HTTP header and the other appears in the URI:

```
GET /product/switch/UID=12345678;ck=1234...  
Host: www.nortelnetworks.com  
Cookie: UID=87654321
```

- Look for the cookie in the HTTP header

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive UID 1 8 dis
```

The last parameter in this command answers the “Look for cookie in URI?” prompt. If you set this parameter to `disable`, the application switch will use `UID=87654321` as the cookie.

- Look for the cookie in the URI

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive UID 1 8 ena
```

The last “Look for cookie in URI?” parameter is set to `enable`. Therefore the application switch will use `UID=12345678` as the cookie.

Example 2: Parsing the Cookie

This example shows three configurations where the switch uses the hashing key or wild cards to determine which part of the cookie value should be used for determining the real server. For example, the value of the cookie is defined as follows:

```
Cookie: sid=0123456789abcdef; name1=value1;...
```

- Select the entire value of the `sid` cookie as a hashing key for selecting the real server:

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive sid 1 16 dis
```

This command directs the switch to use the `sid` cookie, starting with the first byte in the value and using the full 16 bytes.

- Select a specific portion of the `sid` cookie as a hashing key for selecting the real server:

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive sid 8 4 dis
```

This command directs the switch to use the `sid` cookie, starting with the eight byte in the value and using only four bytes. This uses `789a` as a hashing key.

- Using wildcards for selecting cookie names:

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive
    ASPSESSIONID* 1 16 dis
```

With this configuration, the switch will look for a cookie name that starts with `ASPSESSIONID`. `ASPSESSIONID123`, `ASPSESSIONID456`, and `ASPSESSIONID789` will all be seen by the switch as the same cookie name. If more than one cookie matches, only the first one will be used.

Example 3: Using Passive Cookie Mode

If you are using passive cookie mode, the application switch examines the server's `Set-Cookie: value` and directs all subsequent connections to the server that assigned the cookie.

For example, if Server 1 sets the cookie as "`Set-Cookie: sid=12345678`," then all traffic from a particular client with cookie `sid=12345678` will be directed to Server 1.

The following command is used on the application switch:

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive sid 1 8 dis
```

Example 4: Using Rewrite Cookie Mode

- Rewrite server cookie with the encrypted real server IP address:

In cookie rewrite mode, if the cookie length parameter is configured to be eight bytes, the switch will rewrite the placeholder cookie value with the encrypted real server IP address.

```
>> # /cfg/slb/virt 1/service 80/pbind cookie rewrite sid 1 8 dis
```

If the server is configured to include a placeholder cookie, such as follows:

```
Set-Cookie: sid=altheonpersistence;
```

then the application switch will rewrite the first eight bytes of the cookie to include the server's encrypted IP address:

```
Set-Cookie: sid=cdb20f04rsistence;
```

All subsequent traffic from a specific client with this cookie will be directed to the same real server.

- Rewrite server cookie with the encrypted real server IP address and virtual server IP address:

If the cookie length is configured to be 16 bytes, the switch will rewrite the cookie value with the encrypted real server IP address and virtual server IP address.

```
>> # /cfg/slb/virt 1/service 80/pbind cookie rewrite sid 1 16 dis
```

If the server is configured to include a placeholder cookie, as follows:

```
Set-Cookie: sid=altheonwebcookies;
```

then the application switch will rewrite the first 16 bytes of the cookie to include the encrypted real server IP address and virtual server IP address:

```
Set-Cookie: sid=cdb20f04cdb20f0a;
```

All subsequent traffic from a specific client to the particular virtual server IP address with this cookie will be directed to the same real server.

Server-Side Multi-Response Cookie Search

Cookie-based persistence requires the switch to search the HTTP response packet from the server and, if a persistence cookie is found, sets up a persistence connection between the server and the client. The Alteon switch looks through the first HTTP response from the server. While this approach works for most servers, some customers with complex server configurations might send the persistence cookie a few responses later. In order to achieve cookie-based persistence in such cases, Alteon OS allows the network administrator to configure the switch to look through multiple HTTP responses from the server.

In Alteon OS, the network administrator can modify a response counter to a value from 1-16. The switch will look for the persistence cookie in this number of responses (each of them can be multi-frame) from the server.

Configuring Server-Side Multi-Response Cookie Search

Configure the server-side multi-response cookie search by using the following command:

```
>> # /cfg/slb/virt <virtual server>/service <virtual port number>/rcount
Current Cookie search response count:
Enter new Cookie search response count [1-16]:
```

SSL Session ID-Based Persistence

SSL is a set of protocols built on top of TCP/IP that allows an application server and client to communicate over an encrypted HTTP session, providing authentication, non-repudiation, and security. The SSL protocol *handshake* is performed using clear (unencrypted) text. The content data is then encrypted (using an algorithm exchanged during the handshake) prior to being transmitted.

Using the SSL session ID, the switch forwards the client request to the same real server to which it was bound during the last session. Because SSL protocol allows many TCP connections to use the same session ID from the same client to a server, key exchange needs to be done only when the session ID expires. This reduces server overhead and provides a mechanism, even when the client IP address changes, to send all sessions to the same real server.

NOTE – The destination port number to monitor for SSL traffic is user-configurable.

How SSL Session ID-Based Persistence Works

- All SSL sessions that present the same session ID (32 random bytes chosen by the SSL server) will be directed to the same real server.

NOTE – The SSL session ID can only be read by the switch after the TCP three-way handshake. In order to make a forwarding decision, the switch must terminate the TCP connection to examine the request.

- New sessions are sent to the real server based on the metric selected (hash, roundrobin, leastconns, minmisses, response, and bandwidth).
- If no session ID is presented by the client, the switch picks a real server based on the metric for the real server group and waits until a connection is established with the real server and a session ID is received.
- The session ID is stored in a session hash table. Subsequent connections with the same session ID are sent to the same real server. This binding is preserved even if the server changes the session ID mid-stream. A change of session ID in the SSL protocol will cause a full three-way handshake to occur.
- Session IDs are kept on the switch until an idle time equal to the configured server timeout (a default of 10 minutes) for the selected real server has expired.

Figure 18-5 illustrates persistence based on SSL session ID as follows:

1. An SSL Hello handshake occurs between Client 1 and Server 1 via the application switch.
2. An SSL session ID is assigned to Client 1 by Server 1.
3. The application switch records the SSL session ID.
4. The application switch selects a real server based on the existing SLB settings.

As a result, subsequent connections from Client 1 with the same SSL session ID are directed to Server 1.

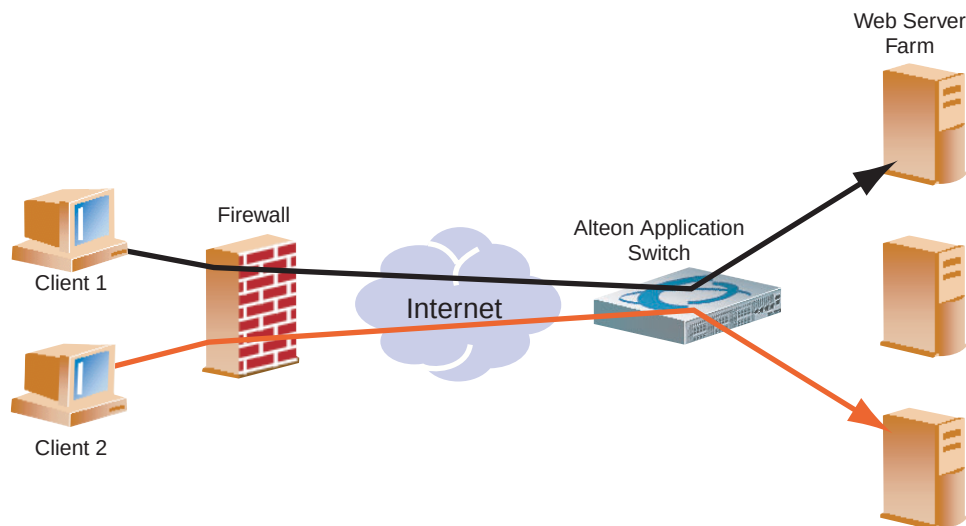


Figure 18-5 SSL Session ID-Based Persistence

5. Client 2 appears to the switch to have the same source IP address as Client 1 because they share the same proxy firewall.

However, the application switch does not automatically direct Client 2 traffic to Server 1 based on the source IP address. Instead an SSL session ID for the new traffic is assigned. Based on SLB settings, the connection from Client 2 is spliced to Server 3.

As a result, subsequent connections from Client 2 with the same SSL session ID are directed to Server 3.

Configuring SSL Session ID-Based Persistence

To configure session ID-based persistence for a real server, perform the following steps:

1. Configure real servers and services for basic SLB, as indicated below:

- Define each real server and assign an IP address to each real server in the server pool.
- Define a real server group and set up health checks for the group.
- Define a virtual server on the virtual port for HTTPS (for example, port 443) and assign a real server group to service it.
- Enable SLB on the switch.
- Enable client processing on the port connected to the client.

For information on how to configure your network for SLB, see [“Server Load Balancing” on page 137](#)

2. If a proxy IP address is not configured on the client port, enable DAM for real servers.

```
>> # /cfg/slb/adv/direct ena
```

3. Select session ID-based persistence as the persistent binding option for the virtual port.

```
>> # /cfg/slb/virt <virtual server number>/service <virtual port> pbind sslid
```

4. Enable client processing on the client port.

```
>> # /cfg/slb/port <port number>/client ena
```




CHAPTER 19

Bandwidth Management

Bandwidth Management (BWM) enables Web site managers to allocate a portion of the available bandwidth for specific users or applications. It allows companies to guarantee that critical business traffic, such as e-commerce transactions, receive higher priority versus non-critical traffic. Traffic classification can be based on user or application information. BWM policies can be configured to set lower and upper bounds on the bandwidth allocation.

The following topics are addressed in this chapter:

- “Enabling BWM on the Switch” on page 492
- “Bandwidth Management Contracts” on page 493
- “Bandwidth Policies” on page 495
- “Data Pacing for Traffic Shaping Contracts” on page 497
- “Rate Limiting Timeslots” on page 498
- “Classification Criteria” on page 498
- “Bandwidth Statistics and History” on page 504
- “Packet Coloring (TOS bits) for Burst Limit” on page 505
- “Configuring Bandwidth Management” on page 506

Enabling BWM on the Switch

To use the bandwidth management features, you must purchase an additional software license and password key. Instructions for obtaining a password key are provided with your software license certificate.

There are two operational keys for BWM: a standard key and a demo key. The demo key automatically expires after a demo time period. These keys may only be enabled if Layer 4 services have been enabled.

Once you have obtained the proper password key to enable BWM, follow these instructions:

- 1. Connect to the switch command line interface via Telnet or the console port, and login as the administrator, following the directions in the *Alteon OS Command Reference*.**
- 2. From the command line prompt, type the `/oper/swkey` command.**

You will be prompted to enter the password key. If the key is correct for this MAC address, the switch will accept the password, permanently record it in the switch's non-volatile RAM (NVRAM), and will then enable the feature.

Bandwidth Management Contracts

To manage bandwidth on an Alteon Application Switch, you must create one or more bandwidth management contracts. The switch uses these contracts to limit individual traffic flows.

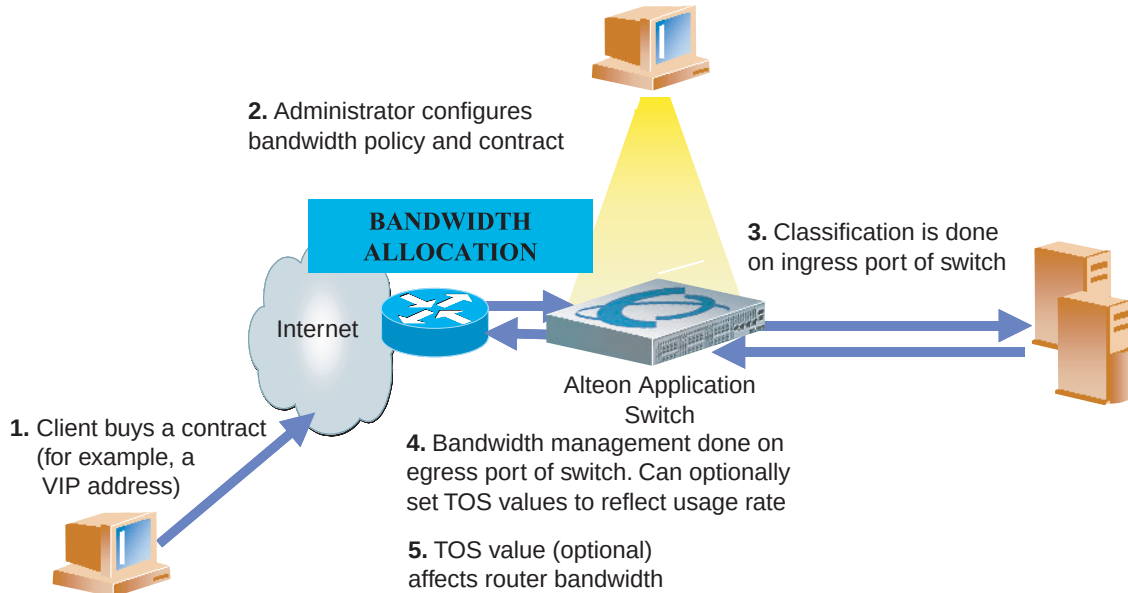


Figure 19-1 Bandwidth Management: How It Works

NOTE – At any given time, up to 256 contracts can be created for a single Alteon Application Switch 2424.

Each contract comprises the following:

- **Classification policy**

In a classification policy certain frames are grouped together. For more information on how frames are associated with a bandwidth management contract, see [“Server Output Bandwidth Control” on page 498](#). For frames qualifying for multiple classifications, precedence of contracts is also specified per contract. If no precedence is specified, the default order is used (see [“Precedence” on page 499](#)).

- **Bandwidth policy**

In a bandwidth policy you specify usage limitations.

■ Traffic classification

Assign contracts to different types of traffic. Traffic can be classified based on layer 2, layer 4, or layer 7 traffic. Traffic classification can be based on the following: port, VLAN, trunk, filters, virtual IP address, service on the virtual server, URL, and so on. When Virtual Matrix Architecture (VMA) is enabled, traffic classification is done on the *ingress port*—that is, the port on which the frame is received (not the *client port* or the *server port*). If the traffic classification is done on layer 4 through layer 7 traffic (filter-based or SLB traffic), then the classification occurs on the *designated port*.

NOTE – For better performance, it is recommended to enable VMA when Bandwidth Management is enabled.

■ Rate Management

Typically, bandwidth management occurs on the egress port of the switch—that is, the port from which the frame is leaving. However, in the case of multiple routes or trunk groups, the egress port can actually be one of several ports (from the point-of-view of where the queues are located).

Rate management for each contract can be done in two ways: Traffic shaping or Rate limiting.

Traffic Shaping. Rate management for a traffic shaping contract is controlled by using queues for each contract and by scheduling when frames are sent from each queue. Each frame is put into a managed buffer and placed on a contract queue. The time that the next frame is supposed to be transmitted for the contract queue is calculated according to the configured rate of the contract, the current egress rate of the ports, and the buffer size set for the contract queue. The scheduler then organizes all the frames to be sent according to their time-based ordering and meters them out to the port.

When packets in a contract queue have not yet been sent and the buffer size set for the queue is full, any new frames attempting to be placed in the queue is discarded.

Rate Limiting. Rate management for a rate limiting contract is controlled by metering the traffic egressing from the switch. If the egress rate is below the configured rate limit (hard limit) for the port the traffic is transmitted immediately without any buffering. If the egress rate is above the configured rate limit the traffic above the rate limit is dropped.

■ Bandwidth policy index

Each BWM contract is assigned a bandwidth policy index and (optionally) a name. This index can be viewed using the `/cfg/bwm/cont` menu. Contracts can be enabled and disabled. The set of classifications associated with each contract can be viewed using the `/info/bwm` menu.

Bandwidth Policies

Bandwidth policies are bandwidth limitations defined for any set of frames, specifying the guaranteed bandwidth rates. A bandwidth policy is often based on a rate structure whereby a Web host or co-location provider could charge a customer for bandwidth utilization. There are three rates that are configured: a Committed Information Rate (CIR)/Reserved Limit, a Soft Limit, and a Hard Limit, as described below.

For traffic shaping contracts, a queue depth is also associated with a policy. A queue depth is the size of the queue that holds the data. It can be adjusted to accommodate delay-sensitive traffic (such as audio) versus drop-sensitive traffic (such as FTP).

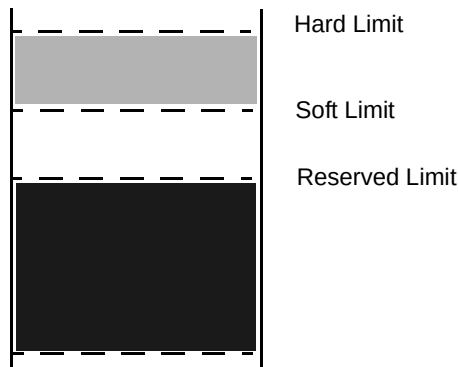


Figure 19-2 Bandwidth Rate Limits

For rate limiting contracts, the queue depth is ignored because traffic is not buffered.

Rate Limits

A bandwidth policy specifies three limits, listed and described in [Table 19-1](#):

Table 19-1 Bandwidth Rate Limits

Rate Limit	Description
Committed Information Rate (CIR) or Reserved Limit	This is a rate that a bandwidth classification is always guaranteed. In configuring BWM contracts, ensure that the sum of all committed information rates never exceeds the link speeds associated with ports on which the traffic is transmitted. In a case where the total CIRs exceed the outbound port bandwidth, the switch will perform a graceful degradation of all traffic on the associated ports.
Soft Limit	For traffic shaping contracts, this is the desired bandwidth rate, that is, the rate the customer has agreed to pay on a regular basis. When output bandwidth is available, a bandwidth class will be allowed to send data at this rate. No exceptional condition will be reported when the data rate does not exceed this limit. For rate limiting contracts, the soft limit is ignored.
Hard Limit	This is a “never exceed” rate. A bandwidth class is never allowed to transmit above this rate. Typically, traffic bursts between the soft limit and the hard limit are charged a premium. The maximum hard limit for a bandwidth policy is 1 Gbps, even when multiple Gigabit ports are trunked.

Bandwidth Policy Configuration

Each bandwidth policy, comprised of the reserved, soft, and hard limits, is assigned an index. These policies can be found under the `/cfg/bwm` menu. Up to 64 bandwidth policies can be defined. Bandwidth limits are usually entered in Mbps.

NOTE – For better granularity, rates can be entered in Kbps by appending “K” to the entered number. For example, 1 Mbps can be entered as either “1” or as “1024k.”

In addition, a queue size is associated with each policy. The queue size is measured in bytes.

Data Pacing for Traffic Shaping Contracts

The mechanism used to keep the individual traffic flows under control is called *data pacing*. It is based on the concept of a virtual clock and theoretical departure times (TDT). The actual calculation of the TDT is based initially on the soft limit rate. The soft limit can be thought of as a target limit for the ISP's customer. So long as bandwidth is available and the classification queue is not being filled at a rate greater than the soft limit, the TDT will be met for both incoming frames and outgoing frames and no borrowing or bandwidth limitation will be necessary.

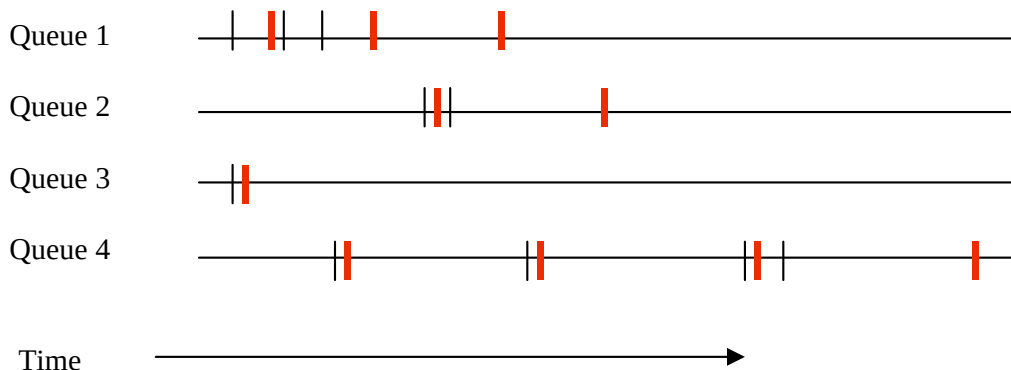


Figure 19-3 Virtual Clocks and TDT

If the data is arriving more quickly than it can be transmitted at the soft limit and sufficient bandwidth is still available, the rate is adjusted upwards based on the depth of the queue, until the rate is fast enough to reduce the queue depth or the hard limit is reached. If the data cannot be transmitted at the soft limit, then the rate is adjusted downward until the data can be transmitted or the CIR is hit. If the CIR is overcommitted among all the contracts configured for the switch, graceful degradation will reduce each CIR until the total bandwidth allocated fits within the total bandwidth available.

BWM configurations will optionally be synchronized during VRRP synchronization except the port contracts and VLAN contracts, which will not be synchronized.

Rate Limiting Timeslots

For rate limiting contracts, metering of individual traffic flows is accomplished using several timeslots per second. The timeslot traffic limit is the traffic that is sent for a particular contract for every timeslot corresponding to the contract's rate limit or the hard limit is initially calculated.

For any contract there is one timeslot traffic limit for each egress port. The timeslot traffic limit is calculated from the hard limit. The timeslot traffic limit is the traffic amount corresponding to the hard limit per second divided by the number of timeslots per second.

Traffic is transmitted for every timeslot, if the traffic to be egressed is below the timeslot traffic limit for the contract. The traffic in excess of the timeslot traffic limit is discarded.

Classification Criteria

The frames associated with a particular BWM contract are specified, using the parameters listed below.

Server Output Bandwidth Control

These classifications are aimed at limiting the traffic outbound from the server farm for bandwidth measurement and control.

- Physical Port - All frames are from a specified physical port.
- VLAN - All frames are from a specified VLAN. If a VLAN translation occurs, the bandwidth policy is based on the ingress VLAN.
- IP Source Address - All frames have a specified IP source address or range of addresses defined with a subnet mask.
- IP Destination Address - All frames have a specified IP destination address or range of addresses defined with a subnet mask.
- Switch services on the virtual servers

The following are various Layer 4 groupings:

- ☐ A single virtual server
- ☐ A group of virtual servers
- ☐ A service for a particular virtual server
- ☐ Select a particular port number (service on the virtual server) within a particular virtual server IP address.

The following are various Layer 7 groupings:

- A single URL path
- A group of URL paths
- A single cookie

Application Bandwidth Control

Classification policies allow bandwidth limitations to be applied to particular applications; that is, they allow applications to be identified and grouped. Classification can be based on any filtering rule, including those listed below:

- TCP Port Number - All frames with a specific TCP source or destination port number
- UDP - All UDP frames
- UDP Port Number - All frames with a specific UDP source or destination port number

Combinations

Combinations of classifications are limited to grouping items together into a contract. For example, if you wanted to have three different virtual servers associated with a contract, you would specify the same contract index on each of the three virtual server IP addresses. You can also combine filters in this manner.

Precedence

If a frame qualifies for different classifications, it is important to be able to specify the classification with which it should be associated. There are two mechanisms to address classification—a per-contract precedence value and a default ordering. If a contract does not have an assigned precedence value, then the ordering is as follows:

- 1. Layer 7 applications (for example, URL, HTTP, headers, cookies, and so forth)**
- 2. Layer 4 services on the virtual server**
- 3. Filter**
- 4. VLAN**
- 5. Source Port/Default Assignment**

URL-Based Bandwidth Management

URL-based BWM allows the network administrator or Web site manager to control bandwidth based on URLs, HTTP headers, or cookies.

All three types of BWM are accomplished by following the configuration guidelines on content switching described in [Chapter 17, “Content Intelligent Switching.”](#) You would also need to assign a contract to each defined string, where the string is contained in a URL, an HTTP header, or a cookie.

BWM based on URLs gives Web site managers the following capabilities:

- Ability to allocate bandwidth based on the type of request

The switch allocates bandwidth based on certain strings in the incoming URL request. For example, as shown in [Figure 19-4](#), if a Web site has 10Mbps of bandwidth, the site manager can allocate 1 Mbs of bandwidth for static HTML content, 3Mbs of bandwidth for graphic content and 4Mbs of bandwidth for dynamic transactions, such as URLs with `cgi-bin` requests and `.asp` requests.

- Ability to prioritize transactions or applications

By allocating bandwidth, the application switch can guarantee that certain applications and transactions get better response time.

- Ability to allocate a certain amount of bandwidth for requests that can be cached

As shown in [Figure 19-5 on page 502](#), users will be able to allocate a certain percentage of bandwidth for Web cache requests by using the URL parsing and bandwidth management feature.

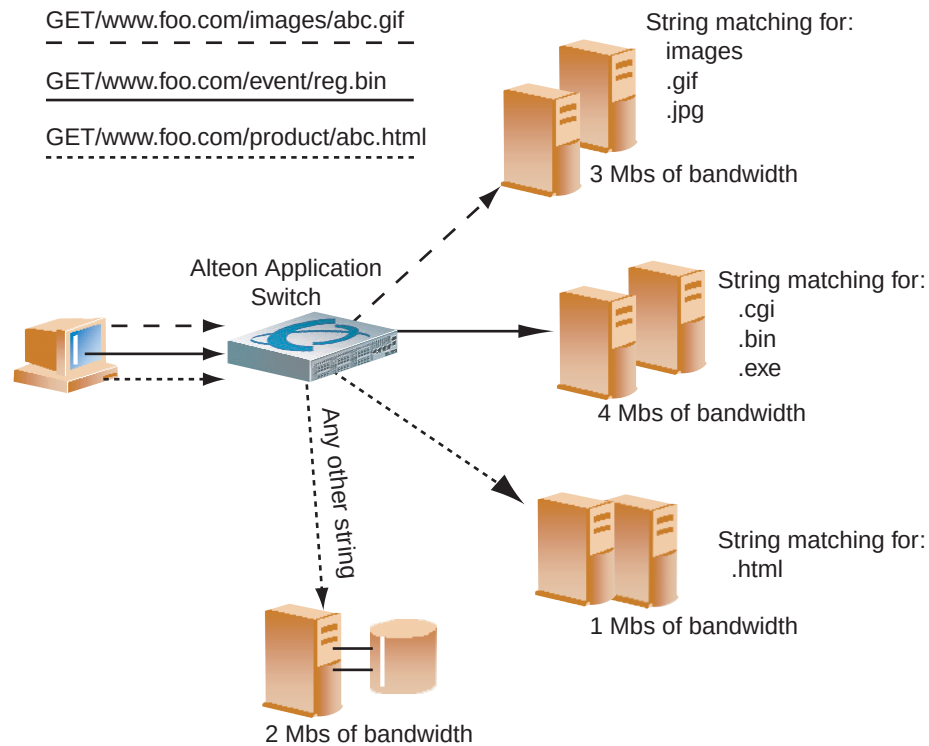


Figure 19-4 URL-Based Bandwidth Management

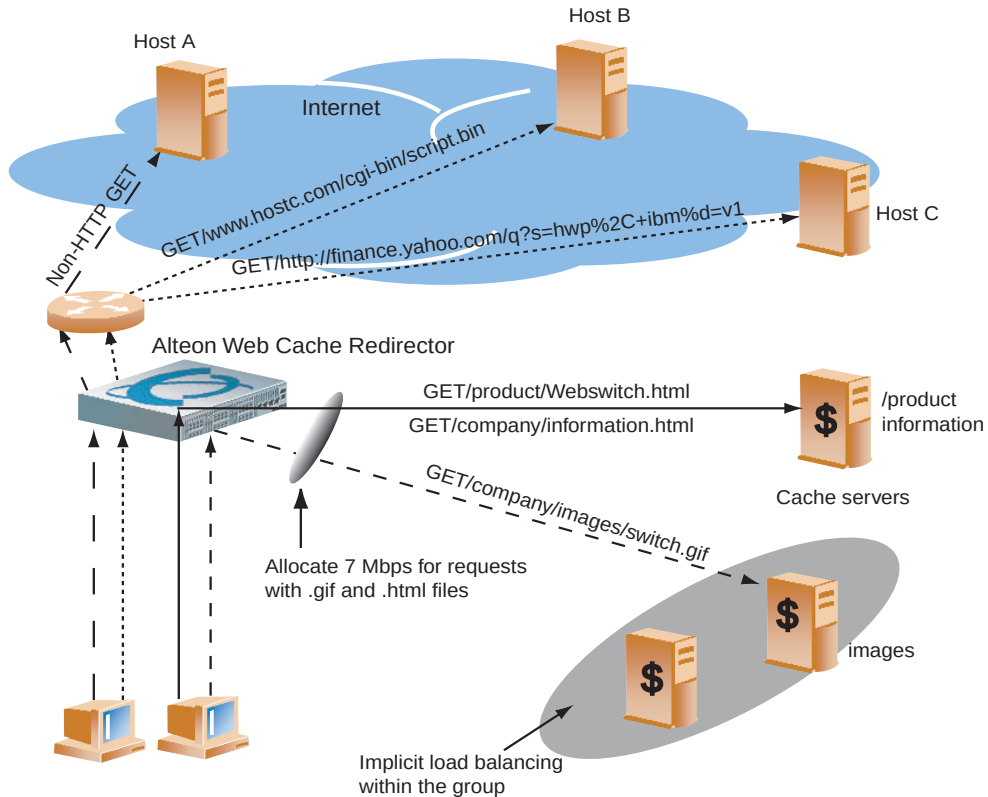


Figure 19-5 URL-Based Bandwidth Management with Cache Redirection

HTTP Header-Based Bandwidth Management

HTTP header-based BWM allows Web site managers to allocate bandwidth based on header value. Thus, they can allocate bandwidth based on browser type, cookie value, and so forth.

Cookie-Based Bandwidth Management

Cookie-based BWM enables Web site managers to prevent network abuse by bandwidth-hogging users. Using this feature, bandwidth can be allocated by type of user or other user-specific information available in the cookie.

Cookie-based bandwidth management empowers service providers to create tiered services. For example, Web site managers can classify users as first class, business class, and coach and allocate a larger share of the bandwidth for preferred classes.

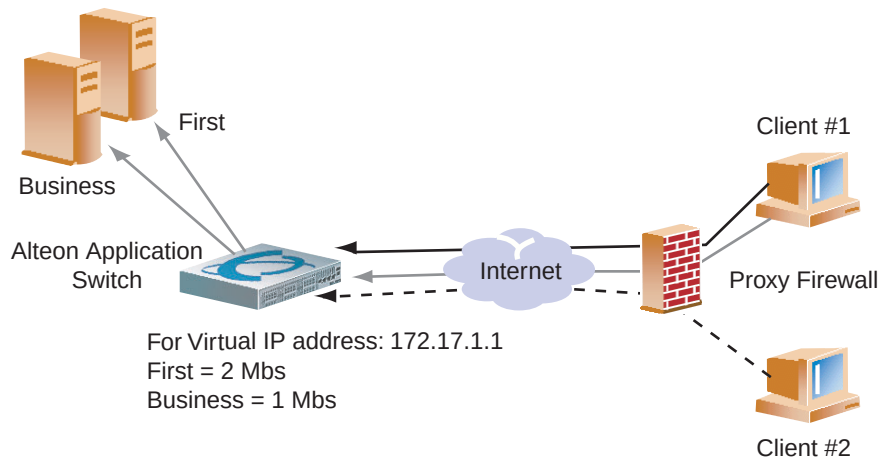


Figure 19-6 Cookie-Based Bandwidth Management

NOTE – Cookie-based BWM does not apply to cookie-based persistency or cookie passive/active mode applications.

Bandwidth Statistics and History

Statistics are maintained in order to allow application switch owners to bill for bandwidth usage. Statistics for frequency and count are configurable. Statistics are kept in the individual Switch Processors (SP) and then collected every second by the MP (Management Processor). The MP then combines the statistics, as statistics for some classifications may be spread across multiple SPs.

The MP maintains some global statistics, such as total octets and a window of historical statistics. When the history buffer of 128K is ready to overflow, it can be optionally e-mailed to a user for long-term storage. Additionally, the history buffer can be sent to the user when the user enters the command: `/oper/bwm/sndhist`. The SMTP protocol is used for this transfer if the SMTP host has been configured (`/cfg/sys/smtp`) and the user e-mail address has been set up (`/cfg/bwm/user`). To obtain graphs, the data must be collected and processed by an external entity through SNMP or through e-mailed logs.

History is maintained only for the contracts for which the history option is enabled (`/cfg/bwm/contract/hist`).

If SMTP is enabled, then the `info` command (`/info/bwm`) can display when the history statistics will be e-mailed to a user.

Statistics Maintained

The total number of octets sent, octet discards, and times over the soft limit are kept for each contract. The history buffer maintains the average queue size for the time interval and the average rate for the interval.

Statistics and Management Information Bases

- For existing BWM classes:
As mentioned above, the MP maintains per-contract rate usage statistics. These are obtainable via a private MIB.
- When BWM services are not enabled:
Even when BWM is not enforced, the MP can still collect classification information and report it, allowing the customer to observe a network for a while before deciding how to configure it. This feature can be disabled using `/cfg/bwm/force dis`. When this command is used, no limits will be applied on any contract.

Packet Coloring (TOS bits) for Burst Limit

Whenever the soft limit is exceeded, optional packet coloring can be done to allow downstream routers to use *diff-serv* mechanisms (that is, writing the Type-Of-Service (TOS) byte of the IP header) to delay or discard these *out-of-profile* frames. Frames that are not out-of-profile are marked with a different, higher priority value. This feature can be enabled or disabled on a per-contract basis, using the `wtos` option under the contract menu (`/cfg/bwm/contract/wtos`) to enable/disable overwriting IP TOS.

The actual values used by the switch for overwriting TOS values (depending on whether traffic is over or under the soft TOS limit) are set in the bandwidth policy menu (`/cfg/bwm/policy/x`) with the `utos` and `otos` options. The values allowed are 0-255. Typically, the values specified should match the appropriate *diff-serv* specification but could be different, depending on the customer environment.

Configuring Bandwidth Management

Any item that is configured with a filter can be used for BWM. Bandwidth classification is performed using the following menus:

- `/cfg/slb/filt` is used to configure classifications based on the IP destination address, IP source address, TCP port number, UDP, UDP port number, or any filter rule.
- `/cfg/slb/virt` is used to configure classifications based on virtual servers.
- `/cfg/port` is used to configure classifications based on physical ports.
(In case of trunking, use `/cfg/trunk`.)
- `/cfg/vlan` is used to configure classifications based on VLANs.
- `/cfg/slb/layer7/lb` is used to configure classification based on URL paths.

To associate a particular classification with a contract, enter the contract index into the “cont” menu option under the applicable configuration menus.

The following procedure provides general instructions for configuring BWM on the switch. Specific configuration examples begin on [page 509](#).

1. Configure the switch as you normally would for SLB. Configuration includes the following tasks:

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Define a real server group.
- Define a virtual server.
- Define the port configuration.

For more information about SLB configuration, see “[Server Load Balancing](#)” on [page 137](#).

2. Enable BWM on the switch.

NOTE – If you purchased the Bandwidth Management option, make sure you enable it by typing `/oper/swkey` and entering its software key.

```
>> # /cfg/bwm/on
```

(Turn BWM on)

3. Select a bandwidth policy.

Each policy must have a unique number from 1 to 64.

```
>> Bandwidth Management# pol 1 (Select bandwidth policy 1)
```

4. Set the hard, soft, and reserved rate limits for the policy, in Mbps.

Typically, charges are applied for burst rates between the soft and hard limit. Each limit must be set between 64K-1000M.

NOTE – For rates less than 1 Mbps, append a “K” suffix to the number.

```
>> Policy 1# hard 6 (Set “never exceed” rate)
>> Policy 1# soft 5 (Set desired bandwidth rate)
>> Policy 1# resv 4 (Set committed information rate)
```

5. (Optional) Set the Type-Of-Service (TOS) byte value, between 0-255, for the policy underlimit and overlimit.

There are two parameters for specifying the TOS bits: underlimit (utos) and overlimit (otos). These TOS values are used to overwrite the TOS values of IP packets if the traffic for a contract is under or over the soft limit, respectively. These values only have significance to a contract if TOS overwrite is enabled in the Bandwidth Management Contract Menu (cfg/bwm/cont x/wtos ena.)

The administrator has to be very careful in selecting the TOS values because of their greater impact on the downstream routers.

```
>> Policy 1# utos 204 (Set BWM policy underlimit)
>> Policy 1# otos 192 (Set BWM policy overlimit)
```

6. Set the buffer limit for the policy.

Set a value between 8192-128000 bytes. The buffer depth for a BWM contract should be set to a multiple of the packet size.

NOTE – Keep in mind that the total buffer limit for the Bandwidth Management policy is 128K.

```
>> Policy 1# buffer 16320 (Set BWM policy buffer limit)
```

7. On the switch, select a BWM contract and (optional) a name for the contract.

Each contract must have a unique number from 1 to 256.

```
>> Policy 1# /cfg/bwm/cont 1           (Select BWM contract 1)
>> BWM Contract 1# name BigCorp        (Assign contract name "BigCorp")
```

8. (Optional) Set a precedence value for the BWM contract.

Each contract can be given a precedence value from 1-256. The higher the number, the higher the precedence. If a frame is applicable to different classifications, then the contract with higher precedence will be assigned to the frame. If the precedence is the same for the applicable contracts, then the following order will be used to assign the contract to the frame:

(1) Incoming port, (2) VLAN, (3) Filter, (4) Service on the Virtual server, (5) URL/Cookie

```
>> BWM Contract 1# prec 1              (Sets contract precedence value to 1)
```

9. (Optional) Enable TOS overwriting for the BWM contract.

```
>> BWM Contract 1# wtos ena            (Enables TOS overwriting for
                                         contract)
```

10. Set the bandwidth policy for this contract.

Each bandwidth management contract must be assigned a bandwidth policy.

```
>> BWM Contract 1# pol 1              (Assign policy 1 to BWM contract 1)
```

11. (optional) Enable traffic shaping.

Rate limiting is enabled by default. Enabling traffic shaping automatically disables rate limiting.

```
>> BWM Contract 1# shaping e          (Enables traffic shaping)
```

12. Enable the BWM contract.

```
>> BWM Contract 1# ena                (Enables this BWM contract)
```

13. Classify the frames for this contract and assign the BWM contract to the filter or virtual IP address.

Each BWM contract must be assigned a classification policy. The classification can be based on a filter or service(s) on the Virtual server. Filters are used to create classification policies based on the IP source address, IP destination address, TCP port number, UDP, and UDP port number.

```
>> BWM Contract 1# /cfg/slb/virt 1/cont 1 (Assign contract to virtual server)
>> Virtual Server 1# /cfg/slb/filt 1/adv/cont 1(Assign contract 1 to filter 1)
```

In this case, all frames that match filter 1 or virtual server 1 will be assigned contract 1.

14. On the switch, apply and verify the configuration.

```
>> Filter 1 Advanced# apply (Make your changes active)
>> Filter 1 Advanced# /cfg/bwm/cur (View current settings)
```

Examine the resulting information. If any settings are incorrect, make any appropriate changes.

15. On the switch, save your new configuration changes.

```
>> Bandwidth Management# save (Save for restore after reboot)
```

16. On the switch, check the BWM information.

```
>> Bandwidth Management# /info/bwm <contract number>(View BWM information)
>> Bandwidth Management# /stats/bwm <contract number>(View BWM statistics)
```

Check that all BWM contract parameters are set correctly. If necessary, make any appropriate configuration changes and then check the information again.

Additional Configuration Examples

Examples are provided for the following Bandwidth Management applications:

- User/Application Fairness: [see next section](#)
- Preferential Services: [page 512](#)
- URL-Based: [page 515](#)
- Cookie-Based: [page 517](#)
- Security Management: [page 520](#)

User/Application Fairness Example

Bandwidth Management can be applied to prevent heavy bandwidth bursters from locking out other users, such as the following:

- Customers using broadband access (such as DSL) blocking dial-up customers
- Customers using the same hosting facility locking out each other because of flash crowd
- FTP locking out Telnet
- Rate limits of particular applications

In the following example, BWM is configured to prevent broadband customers from affecting dial-up customer access. This is accomplished by setting higher bandwidth policy rate limits for the port that processes broadband traffic.

1. Select the first bandwidth policy.

Each policy must have a number from 1 to 64.

NOTE – Ensure BWM is enabled on the switch (`/cfg/bwm/on`).

<pre>>> # /cfg/bwm/pol 1</pre>	<i>(Select BWM policy 1)</i>

2. Set the hard, soft, and reserved rate limits for the bandwidth policy, in Mbps.

<pre>>> Policy 1# hard 5</pre>	<i>(Set “never exceed” rate)</i>
<pre>>> Policy 1# soft 4</pre>	<i>(Set desired bandwidth rate)</i>
<pre>>> Policy 1# resv 3</pre>	<i>(Set committed information rate)</i>

3. On the switch, select a BWM contract and name the contract.

Each contract must have a unique number from 1 to 256.

<pre>>> Policy 1# /cfg/bwm/cont 1</pre>	<i>(Select BWM contract 1)</i>
<pre>>> BWM Contract 1# name dial-up</pre>	<i>(Assign contract name “dial-up”)</i>

4. Set the bandwidth policy for this contract.

Each BWM contract must be assigned a bandwidth policy.

<pre>>> BWM Contract 1# pol 1</pre>	<i>(Assign policy 1 to BWM Contract 1)</i>

5. Enable this BWM contract.

```
>> BWM Contract 1# ena (Enables this BWM contract)
```

6. Select the second bandwidth policy.

```
>> BWM Contract 1# /cfg/bwm/pol 2 (Select bandwidth policy 2)
```

7. Set the hard, soft, and reserved rate limits for this policy, in Mbps.

```
>> Policy 2# hard 30 (Set "never exceed" rate)
>> Policy 2# soft 25 (Set desired bandwidth rate)
>> Policy 2# resv 20 (Set committed information rate)
```

8. On the switch, select the second BWM contract and name the contract.

```
>> Policy 2# /cfg/bwm/cont 2 (Select BWM contract 2)
>> BWM Contract 2# name broadband (Assign contract name "broadband")
```

9. Set the bandwidth policy for this contract.

Each BWM contract must be assigned a bandwidth policy.

```
>> BWM Contract 2# pol 2 (Assign policy 2 to BWM contract 2)
```

10. Enable this BWM contract.

```
>> BWM Contract 2# ena (Enables this BWM contract)
```

11. Assign the BWM contracts to different switch ports.

Physical switch ports are used to classify which frames are managed by each contract—that is, one BWM contract will be applied to all frames from a specific port. The second contract will be applied to all frames from another specified port.

```
>> BWM Contract 2# /cfg/port 1/cont 1 (Assign contract 1 to port 1)
>> Port 1# ../port 2/cont 2 (Assign contract 2 to port 2)
```

12. On the switch, apply and verify the configuration.

>> Port 2# apply	<i>(Make your changes active)</i>
>> Port 2# /cfg/bwm/cur	<i>(View current BWM settings)</i>

Examine the resulting information. If any settings are incorrect, make any appropriate changes.

13. On the switch, save your new configuration changes.

>> Bandwidth Management# save	<i>(Save for restore after reboot)</i>

14. On the switch, check the BWM information.

>> Bandwidth Management# /info/bwm <contract number>	<i>(View BWM information)</i>

Check that all BWM contract parameters are set correctly. If necessary, make any appropriate configuration changes and then check the information again.

Preferential Services Examples

BWM can be used to provide preferential treatment to certain traffic, based on source IP blocks, applications, URL paths, or cookies. You may find it useful to configure higher policy rate limits for specific sites, for example, those used for e-commerce.

Web Site Preference Example

In the following example, there are two Web sites, “A.com” and “B.com.” BWM is configured to give preference to traffic sent to Web site “B.com:”

1. Configure the switch as you normally would for SLB. Configuration includes the following tasks:

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Define a real server group.
- Define a virtual server.
- Define the port configuration.

For more information about SLB configuration, refer to [“Server Load Balancing” on page 137](#).

NOTE – Ensure BWM is enabled on the switch (**/cfg/bwm/on**).

2. Select the first bandwidth policy.

Each policy must have a number from 1 to 64.

```
>> # /cfg/bwm/pol 1
```

(Select BWM policy 1)

3. Set the hard, soft, and reserved rate limits for the bandwidth policy in Mbps.

```
>> Policy 1# hard 10
>> Policy 1# soft 8
>> Policy 1# resv 5
```

(Set “never exceed” rate)
(Set desired bandwidth rate)
(Set committed information rate)

4. On the switch, select a BWM contract and name the contract.

Each contract must have a unique number from 1 to 256.

```
>> Policy 1# /cfg/bwm/cont 1
>> BWM Contract 1# name a.com
```

(Select BWM Contract 1)
(Assign contract name “a.com”)

5. Set the bandwidth policy for this contract.

Each BWM contract must be assigned a bandwidth policy.

```
>> BWM Contract 1# pol 1
```

(Assign policy 1 to BWM contract 1)

6. Enable this BWM contract.

```
>> BWM Contract 1# ena
```

(Enables this BWM contract)

7. Select the second bandwidth policy.

```
>> BWM Contract 1# /cfg/bwm/policy 2
```

(Select BWM policy 2)

8. Set the hard, soft, and reserved rate limits for this policy, in Mbps.

```
>> Policy 2# hard 18
>> Policy 2# soft 15
>> Policy 2# resv 10
```

(Set “never exceed” rate)
(Set desired bandwidth rate)
(Set committed information rate)

9. On the switch, select the second BWM contract and name the contract.

```
>> Policy 2# /cfg/bwm/cont 2           (Select BWM contract 2)
>> BWM Contract 2# name b.com         (Assign contract name "b.com")
```

10. Set the bandwidth policy for this contract.

Each BWM contract must be assigned a bandwidth policy.

```
>> BWM Contract 2# pol 2              (Assign policy 2 to BWM contract 2)
```

11. Enable this BWM contract.

```
>> BWM Contract 2# ena                (Enables this BWM contract)
```

12. Create a virtual server that will be used to classify the frames for contract 1 and assign the Virtual server IP address for this server. Then, assign the BWM contract to the virtual server. Repeat this procedure for a second virtual server.

NOTE – This classification applies to the services within the virtual server and not to the virtual server itself.

The classification policy for these BWM contracts is based on a virtual server. One of the BWM contracts will be applied to any frames that are sent to the virtual server associated with that contract.

```
>> BWM Contract 2# /cfg/slb/virt 1/cont 1 (Assign contract to virtual server 1)
>> Virtual Server 1# vip 100.2.16.2      (Set virtual server VIP address)
>> Virtual Server 1# ena                  (Enable this virtual server)
>> Virtual Server 1# /cfg/slb/virt 2/cont 2 (Assign contract to virtual server)
>> Virtual Server 2# vip 100.2.16.3      (Set virtual server IP address)
>> Virtual Server 2# ena                  (Enable this virtual server)
```

13. On the switch, apply and verify the configuration.

```
>> Virtual Server 2# apply              (Make your changes active)
>> Virtual Server 2# /cfg/bwm/cur      (View current BWM settings)
```

Examine the resulting information. If any settings are incorrect, make the appropriate changes.

14. On the switch, save your new configuration changes.

```
>> Bandwidth Management# save (Save for restore after reboot)
```

15. On the switch, check the bandwidth management information.

```
>> Bandwidth Management# /info/bwm <contract number> (View BWM information)
```

Check that all BWM contract parameters are set correctly. If necessary, make any appropriate configuration changes and then check the information again.

URL-Based Bandwidth Management Example

In this example, you will assign bandwidth based on URL paths. For URL-based server load balancing, a user has to first define strings to monitor. Each of these strings is attached to real servers, and any URL with the string is load balanced across the assigned servers.

NOTE – The complete procedure to configure URL-based SLB is described in Chapter 7, “Content Intelligent Switching” of the *Alteon OS Application Guide*.

The best way to achieve URL-based bandwidth management is to assign a contract to each defined string. This allocates a percentage of bandwidth to the string or URL containing the string.

To configure the switch for URL-based bandwidth management, perform the following steps:

NOTE – Refer to the *Alteon OS Command Reference*, for more information on how to create a contract.

- 1. Define a load-balancing string for URL-based server load balancing. For example, add the string “alteon.”**

```
>> # /cfg/slb/layer7/slb/add alteon
```

To see the URL path ID assigned for the string “alteon,” execute the `cur` command:

```
>> Server Load Balance Resource# cur
Number of entries: 2
1: any, cont 1024
2: alteon, cont 3
```

2. Allocate bandwidth for each string.

To implement this, assign a BWM contract to a defined string.

```
>> Server Load Balance Resource# cont <URL path ID> <BWM Contract number>
```

3. Configure a real server to handle the URL request.

To add a defined string:

```
>> # /cfg/slb/real 2/layer7/addlb <URL path ID>
```

where URL path ID is the identification number of the defined string as displayed when you enter the `cur` command.

Example: `/cfg/slb/real 2/layer7/addlb 3`

4. Either enable Direct Access Mode (DAM) on the switch or configure a proxy IP address on the client port.

NOTE – If VMA is enabled and you are using a proxy IP address, you need to configure proxy IP addresses on ports 1 through 8.

To turn on DAM:

```
>> # /cfg/slb/adv/direct ena
```

To turn off DAM and configure a proxy IP address on the client port:

```
>> # /cfg/slb/adv/direct dis
>> # /cfg/slb/pip
>> Proxy IP Address# pip1 12.12.12.12
>> # /cfg/slb/port 2
>> SLB Port 2# proxy ena
```

For more information on proxy IP addresses, see [“Proxy IP Addresses” on page 156](#).

NOTE – By enabling DAM on the switch or, alternatively, disabling DAM and configuring a proxy IP address on the client port, port mapping for URL-based server load balancing can be performed.

5. Turn on URL-based server load balancing on the virtual server.

Configure everything under the virtual server as in Configuration Example 1.

```
>> # /cfg/slb/virt 1/service 80/httpslb enable urlslb
```

If the same string is used by more than one service, and you want to allocate a certain percentage of bandwidth to this URL string for this service on the virtual server, then define a rule using the `urlcont` command.

```
>> # /cfg/slb/virt 1/service 80/urlcont <URL path ID> <BW Contract number>
```

This contract is tied to service 1. The `urlcont` command will overwrite the contract assigned to the URL string ID.

6. Enable Server Load Balancing.

```
>> # /cfg/slb/on
```

Cookie-Based Bandwidth Management Example

In this example, you will assign bandwidth based on cookies. First, configure cookie-based server load balancing, which is very similar to URL-based load balancing. Any cookie containing the specified string is redirected to the assigned server.

Scenario 1: In this scenario, the Web site has a single virtual server IP address and supports multiple classes of users. Turn on cookie parsing for the service on the virtual server.

```
>> # /cfg/slb/virt 1/service 80
>> # httpslb enabled cookie sid 1 8 disable
```

1. Define one or more load-balancing strings.

```
>> # /cfg/slb/layer7/slb/add <URL path ID>
```

Example:

```
>> # /cfg/slb/layer7/slb/add "Business"
>> # add "First"
>> # add "Coach"
```

2. Allocate bandwidth for each string.

To do this, assign a BWM contract to each defined string.

```
>> # /cfg/slb/layer7/slb/cont <URL path ID> <BWM Contract number>
```

3. Configure a real server to handle the cookie.

To add a defined string:

```
>> # /cfg/slb/real 2/layer7/addlb <URL path ID>
```

where *URL path ID* is the identification number of the defined string.

Example:

```
>> # /cfg/slb/real 2/layer7/addlb 3
```

4. Either enable DAM on the switch or configure a proxy IP address on the client port.

NOTE – If VMA is enabled, you need to configure a unique proxy IP address for each port 1-8.

To turn on DAM:

```
>> # /cfg/slb/adv/direct ena
```

To turn off DAM and configure a Proxy IP address on the client port:

```
>> # /cfg/slb/adv/direct dis
>> # /cfg/slb/pip
>> Proxy IP Address# pip1 12.12.12.12
>> # /cfg/slb/port 2
>> SLB Port 2# proxy ena
```

For more information on proxy IP addresses, see [“Proxy IP Addresses”](#) on page 156.

NOTE – By enabling DAM on the switch or, alternatively, disabling DAM and configuring a proxy on the client port, port mapping for URL-based load balancing can be performed.

5. Enable SLB.

```
>> # /cfg/slb/on
```

Scenario 2: In this scenario, the Web site has multiple virtual server IP addresses, and the same user classification or multiple sites use the same string name. In this scenario, there are two Virtual IP (VIP) addresses: 172.17.1.1 and 172.17.1.2. Both the virtual servers and sites have first class and business class customers, with different bandwidth allocations, as shown in Figure 19-7 on page 519.

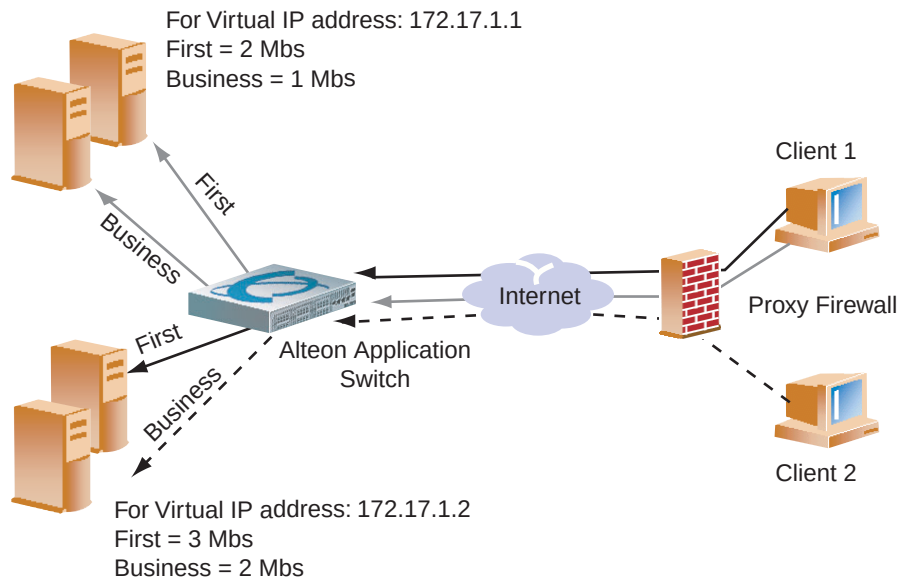


Figure 19-7 Cookie-Based Preferential Services

The configuration to support this scenario is similar to scenario 1. Note the following:

1. **Configure the string and assign contracts for the strings and services.**
2. **If the same string is used by more than one service, and you want to allocate a certain percentage of bandwidth to a user class for this service on the virtual server, then define a rule using the `urlcont` command:**

```
>> # /cfg/slb/virt 1/service 80/urlcont <URL path ID> <BW Contract number>
```

NOTE – When assigning `/cfg/slb/virt 1/service 80/urlcont` (contract 1) and `/cfg/slb/layer7/lb/cont` (contract 2) to the same URL, `urlcont` will override contract 2, even if contract 2 has higher precedence.

Security Management Example

BWM can be used to prevent Denial of Service (DoS) attacks by a flooding of “necessary evil” packets. It can be used to limit the rate of TCP SYN, ping, and other disruptive packets. BWM can also be used to alert the network manager when soft limits are exceeded.

In the following example, a filter is configured to match ping packets, and BWM is configured to prevent DoS attacks by limiting the bandwidth policy rate of those packets:

1. Configure the switch as usual for SLB (see “[Server Load Balancing](#)” on page 137):

- Assign an IP address to each of the real servers in the server pool.
- Define an IP interface on the switch.
- Define each real server.
- Define a real server group.
- Define a virtual server.
- Define the port configuration.

NOTE – Ensure BWM is enabled on the switch (`/cfg/bwm/on`).

2. Select a bandwidth policy.

Each policy must have a number from 1 to 64.

<pre>>> # /cfg/bwm/pol 1</pre>	<i>(Select BWM policy 1)</i>

3. Set the hard, soft, and reserved rate limits for this policy in Kilobytes.

<pre>>> Policy 1# hard 250k</pre>	<i>(Set “never exceed” rate)</i>
<pre>>> Policy 1# soft 250k</pre>	<i>(Set desired bandwidth rate)</i>
<pre>>> Policy 1# resv 250k</pre>	<i>(Set committed information rate)</i>

4. Set the buffer limit for the policy.

Set a parameter between 8192 and 128000 bytes. The buffer depth for a BWM contract should be set to a multiple of the packet size.

<pre>>> Policy 1# buffer 8192</pre>	<i>(Set policy buffer limit of 8192 bytes)</i>

5. On the switch, select a BWM contract and name the contract.

Each contract must have a unique number from 1 to 256.

```
>> Bandwidth Management# /cfg/bwm/cont 1 (Select BWM contract 1)
>> BWM Contract 1# name icmp (Select contract name "icmp")
```

6. Set the bandwidth policy for the contract.

Each BWM contract must be assigned a bandwidth policy.

```
>> BWM Contract 1# pol 1 (Assign policy 1 to BWM contract 1)
```

7. Enable the BWM contract.

```
>> BWM Contract 1# ena (Enables this BWM contract)
```

8. Create a filter that will be used to classify the frames for this contract and assign the BWM contract to the filter.

The classification policy for this BWM contract is based on a filter configured to match ICMP traffic. The contract will be applied to any frames that match this filter

```
>> BW Contract 1# /cfg/slb/filt 1/proto icmp (Define protocol affected by filter)
>> Filter 1# adv/icmp any (Set the ICMP message type)
>> Filter 1 Advanced# cont 1 (Assign BWM contract 1 to this filter)
>> Filter 1 Advanced# /cfg/slb/filt 1/ena (Enable this filter)
>> Filter 1# apply (Port and enable filtering)
```

9. On the switch, apply and verify the configuration.

```
>> Filter 1 Advanced# apply (Make your changes active)
>> Filter 1 Advanced# /cfg/bwm/cur (View current BWM settings)
```

Examine the resulting information. If any settings are incorrect, make the appropriate changes.

10. On the switch, save your new configuration changes.

```
>> Bandwidth Management# save (Save for restore after reboot)
```

11. On the switch, check the BWM information.

```
>> Bandwidth Management# /info/bwm <contract number> (View BWM information)
```

Check that all BWM contract parameters are set correctly. If necessary, make any appropriate configuration changes and then check the information again.

APPENDIX A

Troubleshooting

This section discusses some tools to help you troubleshoot common problems on the Alteon Application Switch:

- [“Monitoring Ports” on page 524](#)
- [“Filtering the Session Dump” on page 526](#)

Monitoring Ports

The port mirroring feature in the Alteon OS allows you to attach a sniffer to a monitoring port that is configured to receive a copy of all packets that are forwarded from the mirrored port. Alteon OS enables you to mirror port traffic for all layers (Layer 2 - 7). Port mirroring can be used as a troubleshooting tool or to enhance the security of your network. For example, an IDS server can be connected to the monitor port to detect intruders attacking the network.

As shown in [Figure A-1](#), port 19 is monitoring *ingress* traffic (traffic entering the switch) on port 3 and *egress* traffic (traffic leaving the switch) on port 11. You can attach a device to port 19 to monitor the traffic on ports 3 and 11.

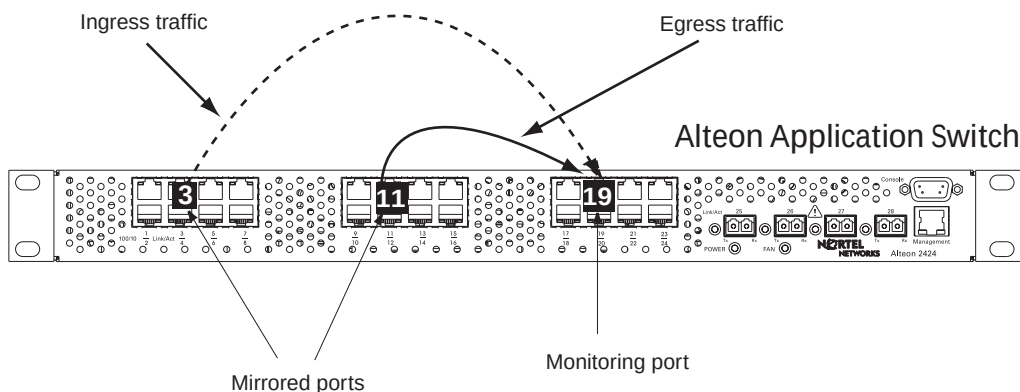


Figure A-1 Monitoring Ports

[Figure A-1](#) shows two mirrored ports monitored by a single port. Similarly, you can have a single or groups of

- a mirrored port to a monitored port
- many mirrored ports to one monitored port

Alteon OS does not support a single port being monitored by multiple ports.

Ingress traffic is duplicated and sent to the mirrored ports *before* processing and egress traffic is duplicated and sent to the mirrored ports *after* processing.

To configure port mirroring for the example shown in [Figure A-1](#),

1. Specify the monitoring port.

```
>> # /cfg/pmirr/monport 19
```

(Select port 19 for monitoring)

2. Select the ports that you want to mirror.

```
>> Port 19 # add 3                                (Select port 3 to mirror)
>> Enter port mirror direction [in, out, or both]: in
                                                    (Monitor ingress traffic on port 3)
>> Port 19 # add 11                                (Select port 11 to mirror)
>> Enter port mirror direction [in, out, or both]: out
                                                    (Monitor egress traffic on port 11)
```

3. Enable port mirroring.

```
>> # /cfg/pmirr/mirr ena                            (Enable port mirroring)
```

4. Apply and save the configuration.

```
>> PortMirroring# apply                            (Apply the configuration)
>> PortMirroring# save                             (Save the configuration)
```

5. View the current configuration.

```
>> PortMirroring# cur                                (Display the current settings)
Port mirroring is enabled
Monitoring Ports      Mirrored Ports
1                     none
2                     none
3                     none
4                     none
5                     none
-----
-----
-----
17                    none
18                    none
19                    (3, in) (11, out)
20                    none
21                    none
```

Filtering the Session Dump

Typically, session dumps are long and unwieldy. Alteon OS however, provides a tool to filter session dumps based on any of the following attributes:

- Client IP address
- Source port
- Destination IP address
- Destination port
- Proxy IP address
- Proxy port
- Matching filter
- Matching flag
- Ingress port
- Real IP address
- SP

Only one attribute can be used at a time to filter the session dump. For example, to filter a session dump based on client IP address 10.10.10.1, use the following command:

```
>> # /info/slb/sess/cip 10.10.10.1
Printing Sessions for SP 1
1,01: 10.10.10.1 137, 205.178.13.100 137 ALLOW age 2 f:100 E
1,01: 10.10.10.1 2592, 172.21.31.100 http -> 14291 10.20.1.2 http age 0
1,01: 10.10.10.1 2593, 172.21.31.100 http -> 14292 10.20.1.3 http age 0
1,01: 10.10.10.1 2594, 172.21.31.100 http -> 14307 10.20.1.2 http age 0
1,01: 10.10.10.1 2595, 172.21.31.100 http -> 14308 10.20.1.3 http age 0
1,01: 10.10.10.1 2596, 172.21.31.100 http -> 14309 10.20.1.4 http age 0
1,01: 10.10.10.1 2597, 172.21.31.100 http -> 14310 10.20.1.3 http age 0
1,01: 10.10.10.1 2598, 172.21.31.100 http -> 14311 10.20.1.4 http age 0
1,01: 10.10.10.1 2599, 172.21.31.100 http -> 14339 10.20.1.3 http age 0
1,01: 10.10.10.1 2600, 172.21.31.100 http -> 14340 10.20.1.4 http age 0
1,01: 10.10.10.1 2601, 172.21.31.100 http -> 14367 10.20.1.3 http age 0
1,01: 10.10.10.1 2602, 172.21.31.100 http -> 14384 10.20.1.2 http age 10
1,01: 10.10.10.1 2603, 172.21.31.100 http -> 14385 10.20.1.3 http age 10
1,01: 10.10.10.1 2604, 172.21.31.100 http -> 14414 10.20.1.2 http age 10

>> Session Table Information#
```

Because VMA is enabled, the command displayed all sessions for client IP address 10.10.10.1 on SP 1. The sessions hash on source IP address and destination IP address.



Glossary

DIP (Destination IP Address)	The destination IP address of a frame.
Dport (Destination Port)	The destination port (application socket: for example, http-80/https-443/DNS-53)
NAT (Network Address Translation)	Any time an IP address is changed from one source IP or destination IP address to another address, network address translation can be said to have taken place. In general, half NAT is when the destination IP or source IP address is changed from one address to another. Full NAT is when both addresses are changed from one address to another. No NAT is when neither source nor destination IP addresses are translated. Virtual server-based load balancing uses half NAT by design, because it translates the destination IP address from the Virtual Server IP address, to that of one of the real servers.
Preemption	In VRRP, preemption will cause a Virtual Router that has a lower priority to go into backup should a peer Virtual Router start advertising with a higher priority.
Priority	In VRRP, the value given to a Virtual Router to determine its ranking with its peer(s). Minimum value is 1 and maximum value is 254. Default is 100. A higher number will win out for master designation.
Proto (Protocol)	The protocol of a frame. Can be any value represented by a 8-bit value in the IP header adherent to the IP specification (for example, TCP, UDP, OSPF, ICMP, and so on.)
Real Server Group	A group of real servers that are associated with a Virtual Server IP address, or a filter.
Redirection or Filter-Based Load Balancing	A type of load balancing that operates differently from virtual server-based load balancing. With this type of load balancing, requests are transparently intercepted and “redirected” to a server group. “Transparently” means that requests are not specifically destined for a Virtual Server IP address that the switch owns. Instead, a filter is configured in the switch. This filter intercepts traffic based on certain IP header criteria and load balances it. Filters can be configured to filter on the SIP/Range (via netmask), DIP/Range (via netmask), Protocol, SPort/Range or DPort/Range. The action on a filter can be Allow, Deny, Redirect to a Server Group, or NAT (translation of either the source IP or destination IP address). In redirection-based load balancing, the destination IP address is not translated to that of one of the real servers. Therefore, redirection-based load balancing is designed to load balance devices that normally operate transparently in your network—such as a firewall, spam filter, or transparent cache.

RIP (Real Server)	Real Server IP Address. An IP addresses that the switch load balances to when requests are made to a Virtual Server IP address (VIP).
SIP (Source IP Address)	The source IP address of a frame.
SPort (Source Port)	The source port (application socket: for example, HTTP-80/HTTPS-443/DNS-53).
Tracking	In VRRP, a method to increase the priority of a virtual router and thus master designation (with preemption enabled). Tracking can be very valuable in an active/active configuration. You can track the following: ■ Vrs: Virtual Routers in Master Mode (increments priority by 2 for each) ■ Ifs: Active IP interfaces on the application switch (increments priority by 2 for each) ■ Ports: Active ports on the same VLAN (increments priority by 2 for each) ■ l4pts: Active Layer 4 Ports, client or server designation (increments priority by 2 for each) ■ reals: healthy real servers (increments by 2 for each healthy real server) ■ hsrp: HSRP announcements heard on a client designated port (increments by 10 for each)
VIP (Virtual Server IP Address)	An IP address that the switch owns and uses to load balance particular service requests (like HTTP) to other servers.
VIR (Virtual Interface Router)	A VRRP address that is an IP interface address shared between two or more virtual routers.
Virtual Router	A shared address between two devices utilizing VRRP, as defined in RFC 2338. One virtual router is associated with an IP interface. This is one of the IP interfaces that the switch is assigned. All IP interfaces on the Alteon Application Switches must be in a VLAN. If there is more than one VLAN defined on the application switch, then the VRRP broadcasts will only be sent out on the VLAN of which the associated IP interface is a member.
Virtual Server Load Balancing	Classic load balancing. Requests destined for a Virtual Server IP address (VIP), which is owned by the switch, are load balanced to a real server contained in the group associated with the VIP. Network address translation is done back and forth, by the switch, as requests come and go. Frames come to the switch destined for the VIP. The switch then replaces the VIP and with one of the real server IP addresses (RIP's), updates the relevant checksums, and forwards the frame to the server for which it is now destined. This process of replacing the destination IP (VIP) with one of the real server addresses is called half NAT. If the frames were not half NAT'ed to the address of one of the RIPs, a server would receive the frame that was destined for it's MAC address, forcing the packet up to Layer 3. The server would then drop the frame, since the packet would contain the destination IP of the VIP and not that of the server (RIP).

VRID (Virtual Router Identifier)

In VRRP, a value between 1 and 255 that is used by each virtual router to create its MAC address and identify its peer for which it is sharing this VRRP address. The VRRP MAC address as defined in the RFC is 00-00-5E-00-01-{VRID}. If you have a VRRP address that two switches are sharing, then the VRID number needs to be identical on both switches so each virtual router on each switch knows whom to share with.

VRRP (Virtual Router Redundancy Protocol)

A protocol that acts very similarly to Cisco's proprietary HSRP address sharing protocol. The reason for both of these protocols is so devices have a next hop or default gateway that is always available. Two or more devices sharing an IP interface are either advertising or listening for advertisements. These advertisements are sent via a broadcast message to an address such as 224.0.0.18.

With VRRP, one switch is considered the master and the other the backup. The master is always advertising via the broadcasts. The backup switch is always listening for the broadcasts. Should the master stop advertising, the backup will take over ownership of the VRRP IP and MAC addresses as defined by the specification. The switch announces this change in ownership to the devices around it by way of a Gratuitous ARP, and advertisements. If the backup switch didn't do the Gratuitous ARP the Layer 2 devices attached to the switch would not know that the MAC address had moved in the network. For a more detailed description, refer to RFC 2338.

VSR (Virtual Server Router)

A VRRP address that is a shared Virtual Server IP address. VSR is Alteon OS proprietary extension to the VRRP specification. The switches must be able to share Virtual Server IP addresses, as well as IP interfaces. If they didn't, the two switches would fight for ownership of the Virtual Server IP address, and the ARP tables in the devices around them would have two ARP entries with the same IP address but different MAC addresses.



Index

Symbols

[]..... 21

Numerics

80 (port) 335
802.1Q VLAN tagging..... 44, 45

A

accessing the switch
 defining source IP addresses..... 33
 RADIUS authentication..... 34
 security..... 32
 using EMS..... 27
 using the Browser-based Interface..... 29
 using the CLI 26
active cookie mode 479
active-active redundancy 294
 configuration..... 304
 Server Load Balancing 306
 synchronization 316
active-standby redundancy 293
 configuration..... 302
administrator account..... 36
aggregating routes 93
 example..... 99
allow (filtering)..... 140, 206
application health checking 267
application ports 148

application redirection..... 241
 client IP address authentication 254
 example with NAT..... 246
 games and real-time applications..... 254
 non cacheable sites 254
 non-HTTP redirects for GSLB 344
 proxies..... 243, 246 to 252
 rport 246, 253
 See cache redirection
 topologies 244
 Web-cache redirection example 242 to 254
authenticating, in OSPF..... 113
authoritative name servers 331
autonomous systems (AS) 106

B

backup servers 155
bandwidth management
 burst limit..... 505
 classification policies..... 498
 configuration, general 506 to 509
 configuration, preferential service 512
 configuration, security 520
 configuration, user fairness 510
 contracts 493, 498
 cookie-based 503
 data pacing 497
 HTTP header-based..... 502
 operational keys 492
 precedence 499
 rate limiting..... 508
 rate limiting timeslots 498
 statistics and history 504
 traffic shaping..... 497, 508
 URL-based..... 500 to 502
 VMA..... 494
bandwidth metric 153

bandwidth policies	495
rates	496
BBI	
See Browser-Based Interface.....	107
Border Gateway Protocol (BGP).....	87
attributes	94
failover configuration.....	96
route aggregation.....	93
route maps	89
selecting route paths.....	95
use with GSLB.....	351
Bridge Protocol Data Unit (BPDU)	63
broadcast domains	43, 45, 48, 79
Browser-Based Interface	107, 261, 297, 335

C

cache redirection	
browser-based	452
delayed binding.....	248
example.....	244
HTTP header-based	451
layer 7 traffic	442
RTSP	249, 456
See application redirection	
servers.....	242 to 244, 245
URL hashing.....	453
URL-based	443
CGI-bin scripts	143, 155
Cisco EtherChannel	59
client proximity tables	348
client traffic processing.....	143, 146
command conventions	21
Command Line Interface.....	26, 107
configuring	
BGP failover	96
cache redirection.....	244
cookie-based persistence.....	480
default filter	209
delayed binding	169
DNS load balancing	174
dynamic NAT.....	229
filter-based security	222
FTP Server Load Balancing	171, 172
IDS load balancing.....	191
IP load balancing	170
IP routing.....	77
multiple services	150
multi-response cookie search.....	486
OSPF	117
port trunking	57
proxy IP addresses	158
RTSP cache redirection	250
server load balancing.....	144
spanning tree groups.....	68
stateful failover.....	324
static NAT	228
TCP rate limiting filters	216
tunable hash for filter redirection.....	220
VLAN-based filters.....	210
WAN link load balancing.....	201
WAP load balancing.....	185
connection time-out	154
console port	26
content intelligent	
cache redirection	442
server load balancing.....	419
contracts, bandwidth management.....	493
cookie	
active	479
different types	477
expiration timer	482
header	475
insert mode	477
names.....	476
passive mode.....	478
permanent	475
temporary	475
values.....	476
cookie-based persistence	474

D

data pacing	497
datagram.....	398
default gateway	76
configuration example	52, 78, 335
VLAN-based.....	49
default password	36
default route	
OSPF	111
delayed binding.....	167 to 169, 417
cache redirection.....	248
deny (filtering)	140, 206
detecting SYN attacks.....	169
dip (destination IP address for filtering).....	214
direct real server access.....	163
Direct Server Return (DSR).....	163
disabling real servers	149
Distributed Site State Protocol (DSSP)	330, 335
dmask	
destination mask for filtering	214
DNS load balancing.....	174
Domain Name System (DNS)	
filtering	221, 224
Global SLB (diagram)	331
load balancing, layer 4.....	172
load balancing, layer 7.....	434
round robin	138
dport (filtering option)	222, 247
DSSP. <i>See</i> Distributed Site State Protocol.	
dynamic NAT	229

E

egress traffic	398
Element Management System (EMS)	27
encrypt	398
EtherChannel	56
as used with port trunking.....	59
expiration timer, insert cookie.....	482
external routing.....	88, 106

F

failed server protection, SLB	138
failover	
active-active.....	299
active-standby	290
overview.....	292
stateful	322

fault tolerance	
port trunking.....	56
Server Load Balancing.....	142
filter redirection	
tunable hash	220
filtering	
configuration example	222
default filter.....	209, 222
IP address ranges	214
layer 7 deny.....	469
NAT configuration example	227 to 229
numbering.....	212
order of precedence	207, 208
proto (option)	222, 247
security example	221
session dumps.....	526
TCP rate limiting	216
filtering-based VLANs	210
firewalls.....	221
fragmenting jumbo frames	74, 76
frame tagging. <i>See</i> VLANs tagging.	
FTP	
applications	148
proxy IP	344
Server Load Balancing.....	171
configuring.....	171, 172

G

gateway. <i>See</i> default gateway.	
Global SLB	
configuration tutorial	334 to 343
Distributed Site State Protocol	330, 335
DNS resolution (diagram)	331
domain name configuration	339
health check interval	338
hostname configuration.....	339
HTTP redirect.....	332
port states.....	337
proximity tables	348
real server groups.....	336
real servers	336
remote site configuration.....	338
tests.....	348
using proxy IP	344
group SLB, metrics	151

H

hash metric	152
hashing	
redirection filters	220
hashing on any HTTP header.....	433
health checks	246, 274
configuration using scripts	263
format	262
Global SLB interval	338
hostname for HTTP content	267 to 268
HTTPS/SSL.....	277
IMAP server	274
RADIUS server.....	276
real server parameters.....	267
real servers	150
script-based.....	261 to 265
SNMP	270
verifying scripts.....	265
wireless session protocol	277 to 279
WSP	278
WTLS	279
high-availability	285
Host routes	
OSPF	116
hostname, for HTTP health checks.....	267, 339
hot-standby redundancy	295
configuration.....	314
HP-OpenView	29
HTTP	
application health checks	267
redirects (Global SLB option)	332
HTTP header	
hashing.....	433
HTTP URL request	468
HTTPS/SSL health checks	277

I

ICMP	207
IDS load balancing	191
IEEE 802.1Q VLAN tagging.....	45
IF. <i>See</i> IP interfaces.	
IGMP.....	207
IMAP server health checks.....	274
imask	149
incoming route maps	90
ingress traffic.....	398

insert cookie mode.....	477
expiration timer	482
internal routing.....	88, 106
Internet Service Provider (ISP), SLB example.....	141
inter-switch port states, for hot-standby redundancy....	296
Intrusion Detection System (IDS).....	188
IP address	
conservation.....	227
filter ranges	214
local route cache ranges	81
private	227
proxies	142, 156, 243, 246 to 252
real server groups.....	145, 308, 336, 448
real servers.....	140, 144, 336
routing example.....	51, 77
SLB real servers	145
virtual servers.....	140, 142, 146, 337
IP interfaces	
configuration example	144, 335
example configuration	51, 77, 80
VLAN #1 (default).....	45
VLANs.....	45
IP proxies	
for application redirection	252
for Global Server Load Balancing	344
for Server Load Balancing	156
<i>See also</i> proxies, proxy IP address (PIP).	
IP routing.....	143
cross-subnet example	74
default gateway configuration.....	52, 78
IP interface configuration.....	51, 77, 80
IP subnets	75
network diagram	75
subnet configuration example	77
switch-based topology	76
IP subnets	76
routing.....	75, 76
VLANs.....	43, 45
ISL Trunking	56

J

jumbo frames	
fragmenting to normal size	74, 76
routing.....	74, 76

L

Layer 4	
administrator account	36
cache redirection	244
server load balancing	144
Layer 7	
cache redirection	442
deny filter	469
precedence lookup	465
regular expression matching	463
server load balancing	419
string matching	461
least connections (SLB Real Server metric)	152
leastconn metric	152
limiting TCP sessions	215
lmask (local route cache parameter)	81
lnet (local route cache parameter)	81
load balancing	
DNS	172, 434
FTP traffic	171
IDS traffic	188
layer 7 traffic	419
RTSP	176
TCP-based DNS queries	172
types of	170 to 198
UDP-based DNS queries	172
WAN traffic	198
WAP traffic	181
local route cache address range	81
local route cache parameters	
lmask	81
lnet	81
log	
filtering option	206
log (filtering option)	221
logical segment. <i>See</i> IP subnets.	
LSAs	105

M

MAC address	400
Main Menu	
Command-Line Interface (CLI)	26
management port	
setting up	31

management port, using	30
Management Processor (MP)	
use in switch security	33
manual style conventions	21
mapping ports	253
mapping virtual ports to real ports	160
matching TCP flags	233
maxcons limit	155
maximum connections	155
metrics	
real server groups	151
MIBs, SNMP	29
minimum misses (SLB real server metric)	151
minmiss metric	151
mirroring ports	524
monitoring ports	524
monitoring real servers	166
multi-homing	351
multi-links between switches	
using port trunking	55
using VLANs	48
multimedia servers	176
multiple services, configuring	150
multiple spanning tree groups	64

N

name servers, Global SLB configuration example	331
Network Address Translation (NAT)	246
configuration example	227 to 229
filter example	230
proxy	230
static example	227
network management	29
network performance	
statistics, with use of proxy addresses	157
NFS server	141
non cacheable sites in application redirection	254
none (port processing mode) example	146
non-HTTP redirects for GSLB	344

O

optional software	241

OSPF	
area types	102
authentication.....	113
configuration examples	118 to 133
default route.....	111
external routes.....	116
filtering criteria	207
host routes	116
link state database.....	105
neighbors.....	105
overview	102
redistributing routes	89, 93
route maps	89, 91
route summarization	110
router ID.....	113
virtual link	112
outgoing route maps	90
overflow servers.....	155
P	
parallel links	48
password	
administrator account	36
default.....	36
L4 administrator account	36
user account.....	36
persistence	
cookie-based	474
multi-reponse cookie search.....	486
SSL session ID-based.....	487 to 489
persistent	
bindings	143
port sessions	325
PIP. <i>See</i> proxies, proxy IP address.	
POP3	344
port 80	335
port mapping	253
port mirroring	524
port processing mode	
client.....	146
none.....	146
server.....	143, 146
port states	337
port trunking.....	56
configuration example.....	57
description	59
EtherChannel	56
fault tolerance	56
ports	
for services.....	148
monitoring	524
physical. <i>See</i> switch ports.	
SLB configuration example.....	146
precedence, in bandwidth classifications	499
private IP address	227
private network	227
protocol types.....	207
proxies	142, 156, 243 to ??, 252
configuration example	230
NAT.....	229
proximity tables.....	348
proxy IP address (PIP)	142, 156, 165, 252
proxy servers.....	243
PVID (port VLAN ID)	44
Q	
QuickTime Streaming Server.....	177
R	
RADIUS	
authentication	34
health checks	276
port 1812 and 1645	148, 183, 185
port 1813	148, 187
server parameters.....	276
SSH/SCP	40
RADIUS snooping.....	184
RADIUS static session entries	183
Rate limiting	496
rate limiting	494, 508
timeslots	498
real server groups	
configuration example	308, 336, 448
real servers.....	142
backup/overflow servers	155
configuration example	336
connection timeouts	154
disable/enable.....	149
health checks	267
maximum connections.....	155
monitoring	166
SLB configuration example.....	145
weights.....	154
redirect (HTTP).....	332

redirecting filters	
tunable hash	220
redirecting non-HTTP applications.....	344
redirection. <i>See</i> application redirection	
redistributing routes.....	89, 93, 99
redundancy	
active-active.....	294
active-standby	293
hot-standby	295
remote (Global SLB real server property)	338
response metric	153
RIP (Routing Information Protocol)	
advertisements.....	86
distance vector protocol	85
hop count.....	85
metric.....	85
routing table.....	86
TCP/IP route information.....	20, 85
UDP.....	86
version 1.....	85
roundrobin	
SLB Real Server metric	152
route aggregation.....	93, 99
route cache, sample	50
route maps	89
configuring	91
incoming and outgoing	90
route paths in BGP	95
Router ID	
OSPF	113
routers	52, 75, 78
border.....	106
peer.....	106
port trunking	56
switch-based routing topology	76
using redirection to reduce Internet congestion	241
web-cache redirection example.....	242

routes, advertising.....	106
routing.....	88
internal and external	106
Routing Information Protocol. <i>See</i> RIP	
rport (filtering)	246, 253
RSA keys.....	39
RTSP	
cache redirection	249
server load balancing.....	176

S

scalability, service	138
SCP	
services.....	39
script-based health checks	261 to 265
searching for cookie.....	482
searching for cookies	482
SecurID	41
security	
allowable SIP addresses	33
filter-based	221
filtering.....	206, 221
firewalls.....	221
from viruses	206
layer 7 deny filter	468
port mirroring.....	524
private networks.....	227
RADIUS authentication	34
switch management	33
VLANs.....	43
segmentation. <i>See</i> IP subnets.	
segments. <i>See</i> IP subnets.	
server (port processing mode) example	146
server failure	283

Server Load Balancing	
across subnets	74
active-active redundancy	306
backup servers	155
complex network topologies	156
configuration example	141, 156
direct real server access	163
distributed sites	330
DNS	172, 434
failed server protection	138
fault tolerance	142
FTP	171
health checks	267
IDS	188
maximum connections	155
metrics	151
overflow servers	155
overview	139
persistent bindings	143
port processing modes	143, 146
proxies	142, 156
proxy IP addresses	165
real server group	145, 308, 448
real server IP address (RIP)	140
real servers	142
remote sites	330
RTSP	176
topology considerations	142
virtual IP address (VIP)	140, 142
virtual servers	140, 146
WAN links	198
WAP	181
weights	154
server pool	138
server port processing	143
service failure	283
service ports	148
session dumps	526
setting multiple response count	482
shared services	138
SIP (source IP address for filtering)	214
smask	
source mask for filtering	214
SMTP	344
SNMP	29, 107
assign health check	261
HP-OpenView	29
MIBS	29
SNMP content health check	270
SNMP health check	270
Spanning-Tree Protocol	
eliminating bridge loops with VRRP	318
multiple instances	64
replacing with hot-standby failover	314
VLANs	48
spoofing, prevention of	33
sport (filtering option)	222, 247
SSH	
RSA host and server keys	39
SSH/SCP	
configuring	41
stateful failover	322
static NAT	227
static session entries	183
statistical load distribution	56
streaming media	176
summarizing routes	110
switch failover	292
switch management	
security	33
via IP interface	45
switch ports VLANs membership	44
syslog	
messages	206

T

tagging. <i>See</i> VLANs tagging.	
TCP	207, 224, 225
health checking using	150
port 80	165
rate limiting	215
TCP flags	233
TCP/UDP port numbers	160
TDT (Theoretical Departure Times)	497
Telnet	221
text conventions	21
Theoretical Departure Times (TDT)	497
timeouts for real server connections	154
TOS burst limit for bandwidth management	505
Traffic shaping	496, 497
traffic shaping	494, 497, 508
transparent proxies	156, 243, 246 to 252
troubleshooting	526
tunable hashing	220
tunneling	398
typographic conventions	21

U

UDP	207, 224, 225
jumbo frame traffic fragmentation	76
RIP	86
server status using	150
user account	36
Using proxy IP	
GSLB	344

V

virtual clocks	497
virtual interface router (VIR)	286
virtual IP address (VIP)	140, 142
virtual link, OSPF	112
Virtual Local Area Networks. <i>See</i> VLANs.	
Virtual Matrix Architecture (VMA)	156
virtual port mapping to multiple real ports 160 to 163	
virtual router	
ID numbering	319
virtual router group	296
Virtual Router Redundancy Protocol	
tracking	300
virtual server router	298
virtual servers	140
configuration example	146
IP address	146, 337
virus attacks, preventing	468
VLAN-based filtering	210

VLANs

broadcast domains	43, 45, 48, 79
default PVID	44
example showing multiple VLANs	46
filtering	210
gateway, default	49
ID numbers	44
IP interface configuration	80
IP interfaces	45
multiple links	48
multiple spanning trees	62
multiple VLANs	44, 45
parallel links example	48
port members	44
PVID	44
routing	79
security	43
Spanning-Tree Protocol	48, 62
tagging	44 to 47
topologies	45
VLAN #1 (default)	44, 45, 145, 245, 335
VLAN-tagging adapter support for	45
VPN	397
VPN cluster	398
VPN Load Balancing overview	398
VRRP (Virtual Router Redundancy Protocol)	
active-active redundancy	294
active-standby redundancy	293
hot-standby redundancy	295
inter-switch port states	296
overview	286, 300
synchronization	316
synchronizing configurations	296
virtual interface router	286
virtual router ID numbering	319
virtual server router	298
vrid	287

W**WAP**

WSP content health check	278
WTLS health check	279
WAP Gateway	181
WAP load balancing	
RADIUS snooping	184
static session entries	182
Web cache redirection	
<i>See</i> application redirection	

Web hosting	141	Wireless Application Protocol	181
weights	154	World Wide Web, client security for browsing	221
Wide Area Networking (WAN)		WSP health check.....	278
inbound traffic.....	199	WSP health checks	277 to 279
load balancing	198	WTLS health check	279
outbound traffic.....	199		