

Web OS Switch Software



8.3 Application Guide

Part Number: 050131, Revision A, January 2001



50 Great Oaks Boulevard
San Jose, California 95119
408-360-5500 Main
408-360-5501 Fax
www.alteonwebsystems.com

Copyright 2001 Alteon WebSystems, Inc., 50 Great Oaks Boulevard, San Jose, California 95119, USA. All rights reserved. Part Number: 050131, Revision A.

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 Alteon WebSystems, 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).

Alteon WebSystems, Inc. reserves the right to change any products described herein at any time, and without notice. Alteon WebSystems, Inc. assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by Alteon WebSystems, 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 Alteon WebSystems, Inc.

Web OS, Alteon 180, ACEdirector, and ACEswitch are trademarks of Alteon WebSystems, 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.

Contents

Preface 17

Who Should Use This Guide 17

What You'll Find in This Guide 17

Typographic Conventions 18

Contacting Alteon WebSystems 19

Chapter 1: Server Load Balancing 21

Overview 21

 Benefits 21

 Identifying Your Network Needs 22

 How SLB Works 22

Network Topology Considerations 24

Virtual Matrix Architecture 26

 Configuration Issues 26

SLB Implementation 27

 Web Hosting Configuration 27

Additional SLB Options 33

 IP Address Ranges Using imask 33

 Health Checks for Real Servers 33

 Metrics for Real Server Groups 34

 Weights for Real Servers 36

 Connection Time-Outs for Real Servers 36

 Maximum Connections for Real Servers 36

 Backup/Overflow Servers 37

 Mapping Virtual Ports to Real Ports 37

 Direct Server Return 38

 Proxy IP Addresses for Complex SLB Networks 39

Direct Client Access to Real Servers 42

 Direct Access Mode 42

 Multiple IP Addresses on the Server 42

 Proxy IP Addresses 42

 Port Mapping 43

 Management Network 44

FTP Server Load Balancing	44
Background	45
Configuring FTP SLB	45
Mapping a Single Virtual Port to Multiple Real Ports	46
Using the /cfg/slb/virt Command	48
Using the /cfg/slb/real Command	49

Chapter 2: Filtering 51

Overview	51
Benefits	51
Filtering Criteria	52
Stacking Filters	53
Overlapping Filters	53
The Default Filter	54
Numbering Filters	54
Filter Logs	55
Security Example	56
Example Configuration for the Security Solution	57
TCP ACK Matching for Filters	62
Option to List ICMP Filtering Types	65
Network Address Translation Examples	66
Internal Client Access to Internet	66
External Client Access to Server	67
Defining IP Address Ranges for Filters	69
Web OS 8.3 Filtering Additions	70
Full TCP Flag Filtering	70
ICMP Type Filtering	72
FTP Client NAT (Active FTP for Dynamic NAT)	74
Configuring Active FTP Client NAT	75

Chapter 3: Application Redirection 77

Overview	77
Web Cache Redirection Environment	78
Example Web-Cache Solution Configuration	79
IP Proxy Addresses	84
Excluding Non-Cacheable Sites	86
Defining IP Address Ranges for Filters	87
Additional Application Redirection Options	87

Chapter 4: Firewall Load Balancing 89

Overview	89
Basic FWLB Implementation	90
Configuration for Basic FWLB - an Example	92
Adding a DMZ	99
Firewall Health Checks	101
Firewall Service Monitoring	101
Physical Link Monitoring	101
Using HTTP Health Checks	102
Spanning Tree	102
FWLB Checklist	103
Primary Switch	103
Secondary Switch	103
Firewall Configuration Examples	104
Four Subnet FWLB Using Alteon WebSwitches	104
4 WebSwitches, 2 Routing Firewalls, with No Interior Hubs	110
Configuring NAT on Solaris Firewalls	117
NAT for Single Devices on the Clean-Side Network	118
Static NAT, Performed by Firewall	122

Chapter 5: Virtual Private Network Load Balancing 125

Overview	125
Virtual Private Networks	125
How VPN Load Balancing Works	125
VPN Load-Balancing Configuration	127
Requirements	127
VPN Load Balancing Configuration Example	128

Chapter 6: Global Server Load Balancing 145

GSLB Overview	145
Benefits	146
How GSLB Works	147
GSLB Configuration Example	149
Summary	149
Example GSLB Configuration Procedure	150
IP Proxy Addresses for Non-HTTP Application Redirects	161
Basic Tests for GSLB Operation	163
GSLB Client Proximity Tables	164
GSLB Proximity Configuration Example	165

Chapter 7: Content Intelligent Switching 167

Content Switching Overview	168
URL-Based Web Cache Redirection	170
Configuring URL-Based Web-Cache Redirection	172
Statistics for URL-Based WCR	178
URL-Based Server Load Balancing	179
Configuring URL-Based SLB	180
Statistics for URL-Based SLB	183
HTTP Header Inspection	184
Multiple Frames Processing for Delayed Binding	184
HTTP Header-Based SLB	185
No Cache/Cache Control for WCR	185
Configuring HTTP Header-Based Web-Cache Redirection	185
Configuring Browser-Based WCR	188
Virtual Hosting	189
Virtual Hosting Configuration Overview	190
Configuring the “Host:” Header for Virtual Hosting	191
Browser-Smart Load Balancing	192
Configuring Browser-Based Load Balancing	192
Cookie-Based Preferential Load Balancing	193
Configuring Cookie-Based Preferential Load Balancing	194
URL Hashing	196
URL Hashing for Web-Cache Redirection	196
URL Hashing for Server Load Balancing	196
Configuring URL Hashing	198
Exclusionary String Matching for URL SLB	199
Configuring Exclusionary URL Substring Matching	200

Chapter 8: Persistence 203

IP Source Address-Based Persistence	203
Cookie-Based Persistence	204
Types of HTTP Cookies	205
Modes of Operation	206
Cookie Assignment Servers	208
Configuring Cookie-Based Persistence	212
Directing Cookie Client to a Specific Server	216
SSL Session ID-Based Persistence	218
Server-Side Multi-Response Cookie Search	221
Configuring Server-Side Multi-Response Cookie Search	221

Chapter 9: Bandwidth Management 223

Overview	223
Bandwidth Policies	225
Rate Limits	226
Bandwidth Policy Configuration	226
Data Pacing	227
Classification Criteria	228
Server Output Bandwidth Control	228
Application Bandwidth Control	228
Combinations	229
Precedence	229
Bandwidth Classification Configuration	229
Frame Discard	230
URL-Based Bandwidth Management	230
HTTP Header-Based Bandwidth Management	232
Cookie-Based Bandwidth Management	232
Bandwidth Statistics and History	233
Statistics Maintained	233
Statistics and Management Information Bases	233
Packet Coloring (TOS bits) for Burst Limit	234
Operational Keys	234
Configuring Bandwidth Management	235
Additional Configuration Examples	238
Preferential Services Examples	241

Chapter 10: Health Checking 251

Health-Check Parameters for Real Servers	251
Health-Check Types	252
TCP Health Checks	252
ICMP Health Checks	252
Wireless Session Protocol Content Health Checks	252
Hostname for HTTP Content Health Checks	255
IMAP Server Health Checks	256
RADIUS Server Health Checks	257
Script-Based Health Checks	258
HTTPS/SSL Health Check	263
Failure Types	264
Service Failure	264
Server Failure	264

Chapter 11: Secure Switch Management 265

- Secure Switch Management 266
 - Authentication and Authorization 266
- RADIUS Authentication 268
 - RADIUS Authentication Features in Web OS 269
- Secure Shell (SSH) and Secure Copy (SCP) 272
 - Encryption of Management Messages 273
 - SCP Services 274
 - RSA Host and Server Keys 275
 - Radius Authentication 275
 - SecurID Support 276

Chapter 12: High Availability 279

- Failover Methods: An Overview 280
 - Active-Standby Redundancy 281
 - Active-Active Redundancy 282
 - Hot-Standby Redundancy 282
 - Configuration Synchronization 285
- VRRP Overview 286
 - VRRP Components 286
 - VRRP Operation 288
 - Determining Which VRRP Router Is the Master 288
 - Active-Standby Failover 289
- Alteon Extensions to VRRP 290
 - Virtual Server Routers 290
 - Sharing/Active-Active Failover 291
 - Tracking 292
- Redundancy Configurations 294
 - Active-Standby Virtual Server Router Configuration 294
 - Active-Active VIR and VSR Configuration 296
 - Active/Active Server Load Balancing Configuration 298
 - Hot-Standby Configuration 306
- Virtual Router Deployment Considerations 308
 - Mixing Active-Standby and Active-Active Virtual Routers 308
 - VRRP Active/Active Synchronization 308
 - Using the /oper/slb/sync Command 309
 - Using the /cfg/slb/sync Command 309
 - VRRP, STP, and Failover Response Time 310
 - VRRP Virtual Router ID Numbering 311
 - Configuring Tracking 311

Stateful Failover of L4 and L7 Persistent Sessions	313
Overview	313
What Happens When a Switch Fails	314
Stateful Failover Configuration Example	315
Viewing Statistics on Persistent Port Sessions	316

Chapter 13: VLANs 317

VLAN ID Numbers	317
VLAN Tagging	318
VLANs and Spanning-Tree	318
VLANs and the IP Interfaces	318
VLAN Topologies and Design Issues	319
Example 1: Multiple VLANs with Tagging Adapters	319
Example 2: Parallel Links with VLANs	321

Chapter 14: Jumbo Frames 323

Isolating Jumbo Frame Traffic using VLANs	323
Routing Jumbo Frames to Non-Jumbo Frame VLANs	324

Chapter 15: IP Routing 325

IP Routing Benefits	325
Example of Routing Between IP Subnets	325
Defining IP Address Ranges for the Local Route Cache	332
Border Gateway Protocol (BGP)	333
Internal Routing vs. External Routing	333
BGP Failover Configuration	334
BGP-Based Global Server Load Balancing	337
DHCP Relay	337
DHCP Overview	337
DHCP Relay Agent Configuration	338

Chapter 16: Port Trunking 341

Port Trunking Overview	341
Basics	341
Statistical Load Distribution	342
Built-In Fault Tolerance	342
Port Trunking Example	343

Glossary 345

Index 349



Figures

Figure 1-1: Traditional Versus SLB Network Configurations	23
Figure 1-2: SLB Client/Server Traffic Routing	24
Figure 1-3: Example Network for Client/Server Port Configuration	25
Figure 1-4: Web Hosting Configuration without SLB	27
Figure 1-5: Web Hosting with SLB Solutions	28
Figure 1-6: Direct Server Return	38
Figure 1-7: Mapped and Non-Mapped Server Access	43
Figure 1-8: Basic Virtual Port to Real Port Mapping Configuration	47
Figure 2-1: Assigning Filters According to Range of Coverage	53
Figure 2-2: Assigning Filters to Overlapping Ranges	53
Figure 2-3: Assigning a Default Filter	54
Figure 2-4: Example Security Topology	56
Figure 2-5: Example Filter TCP ACK Matching Network	62
Figure 2-6: Dynamic NAT	66
Figure 2-7: Static NAT	67
Figure 2-8: Many to Many NAT	74
Figure 3-1: Traditional Network Without Web Cache Redirection	78
Figure 3-2: Network with Web Cache Redirection	78
Figure 4-1: Typical Firewall Configuration Before FWLB	89
Figure 4-2: Basic FWLB Topology	90
Figure 4-3: Typical Firewall Load-Balancing Topology with DMZ	99
Figure 4-4: Four-Subnet FWLB with Alteon Switches	104
Figure 4-5: 4 WebSwitches, 2 Routing Firewalls, with No Interior Hubs	110
Figure 4-6: NAT for Single Devices on Clean-Side Network	118
Figure 5-1: Basic Network Frame Flow and Operation	126
Figure 5-2: VPN Load-Balancing Configuration Example	128
Figure 5-3: Checkpoint Rules for Both VPN Devices as Seen in the Policy Editor	141

Figure 6-1: DNS Resolution with Global Server Load Balancing	147
Figure 6-2: GSLB Example Topology	150
Figure 6-3: POP3 Request Fulfilled via IP Proxy	161
Figure 6-4: GSLB Proximity Tables: How They Work	164
Figure 7-1: Content-Aware Load Balancing Example	169
Figure 7-2: URL-Based Web-Cache Redirection	171
Figure 7-3: URL-Based Server Load Balancing	179
Figure 7-4: Balancing Non-Transparent Caches	197
Figure 8-1: Cookie-Based Persistence: How It Works	204
Figure 8-2: Passive Cookie Mode	206
Figure 8-3: Active Cookie Mode	207
Figure 8-4: SSL Session ID-Based Persistence	219
Figure 9-1: Bandwidth Management: How It Works	223
Figure 9-2: Bandwidth Rate Limits	225
Figure 9-3: Virtual Clocks and TDT	227
Figure 9-4: URL-Based Bandwidth Management	231
Figure 9-5: URL-Based Bandwidth Management with Web Cache Redirection	231
Figure 9-6: Cookie-Based Bandwidth Management	232
Figure 9-7: Cookie-Based Preferential Services	248
Figure 11-1: Authentication and Authorization: How It Works	268
Figure 11-2: Secure Switch Management: How It Works	273
Figure 12-1: A Non-VRRP, Hot-Standby Configuration	280
Figure 12-2: Active-Standby Redundancy	281
Figure 12-3: Active-Active Redundancy	282
Figure 12-4: Hot-Standby Redundancy	283
Figure 12-5: VRRP Router Example 1	287
Figure 12-6: VRRP Router Example 2	289
Figure 12-7: Active-Active High Availability	291
Figure 12-8: Active-Standby High Availability Configuration	294
Figure 12-9: Active-Active High-Availability Configuration	296
Figure 12-10: Hot-Standby Configuration	306
Figure 12-11: Loops in Active-Active Configuration	310
Figure 12-12: Cross-Redundancy Creates Loops, But STP Resolves Them	310
Figure 12-13: VLANs can be used to Create Non-Looping Topologies	311
Figure 12-14: Stateful Failover Example when the Master Switch Fails	314

Figure 13-1: Example #1: Multiple VLANs with Tagging ACEnic Adapters 319

Figure 13-2: Example 2: Parallel Links with VLANs 321

Figure 14-1: Jumbo Frame VLANs 324

Figure 15-1: The Router Legacy Network 326

Figure 15-2: Switch-Based Routing Topology 327

Figure 15-3: BGP Configuration Example 334

Figure 15-4: DHCP Relay Agent Configuration 338

Figure 16-1: Port Trunk Group 341

Figure 16-2: Port Trunk Group Configuration Example 343



Tables

Table 1-1: Web Host Example: Real Server IP addresses 29

Table 1-2: Web Host Example: Port Usage 31

Table 1-3: Proxy Example: Alteon 180 Port Usage 40

Table 2-1: Well-Known Protocol Types 52

Table 2-2: Well-Known Application Ports 52

Table 2-3: Web-Cache Example: Real Server IP addresses 57

Table 2-4: Filtering IP Address Ranges 69

Table 2-5: TCP Flags 70

Table 2-6: ICMP Message Types 72

Table 3-1: Web Cache Example: Real Server IP addresses 79

Table 3-2: Filtering IP Address Ranges 87

Table 6-1: GSLB Example: California Real Server IP Addresses 152

Table 6-2: GSLB Example: California Alteon 180 Port Usage 153

Table 6-3: Denver Real Server IP Addresses 157

Table 6-4: Web Host Example: Alteon 180 Port Usage 158

Table 9-1: Bandwidth Rate Limits 226

Table 9-2: Bandwidth Policy Limits 226

Table 11-1: User Access Levels 270

Table 11-2: Alteon WebSystems User Access Levels 271

Table 12-1: VRRP Tracked Parameters 292

Table 15-1: Subnet Routing Example: IP Address Assignments 328

Table 15-2: Subnet Routing Example: IP Interface Assignments 328

Table 15-3: Subnet Routing Example: Optional VLAN Ports 330

Table 15-4: Local Routing Cache Address Ranges 332



Preface

This *Application Guide* describes how to configure and use the Web OS 8.3 software included in the Alteon WebSystems family of 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, the IEEE 802.1d Spanning Tree Protocol, and SNMP configuration parameters.

What You'll Find in This Guide

The chapters in this guide will help you plan, implement, and administer the use of Web OS software features. Where possible, each chapter provides a conceptual overview of a specific Web OS feature or functional area, usage examples, and, configuration instructions for implementing the feature(s) on an Alteon WebSystems switch.

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 <IP address> 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]

Contacting Alteon WebSystems

Use the following information to access Alteon WebSystems' support and sales.

- URL for Alteon WebSystems Online:

<http://www.alteonwebsystems.com>

This website includes product information, software updates, release notes, and white papers. The website also includes access to Alteon WebSystems Customer Support for accounts under warranty or covered by a maintenance contract.

- E-mail access:

support@alteonwebsystems.com

E-mail access to Alteon WebSystems Customer Support is available for accounts under warranty or covered by a maintenance contract.

- Telephone access to Alteon WebSystems Customer Support:

1-888-Alteon0 (or 1-888-258-3660)

1-408-360-5695

Telephone access to Alteon WebSystems Customer Support is available to accounts that are under warranty or covered by a maintenance contract. Normal business hours are 8 a.m. to 6 p.m. Pacific Standard Time.

- Telephone access to Alteon WebSystems Sales:

1-888-Alteon2 (or 1-888-258-3662), and press 2 for Sales

1-408-360-5600, and press 2 for Sales

Telephone access is available for information regarding product sales and upgrades.



CHAPTER 1

Server Load Balancing

This chapter describes how to configure and use the optional Layer 4 software for Server Load Balancing (SLB). For information on activating this optional software, see your *Web OS 8.3 Command Reference*.

Overview

Benefits

SLB benefits your network in a number of ways:

- Increased efficiency for server utilization and network bandwidth

Your Alteon Web OS switch is aware of the shared services provided by your server pool. The switch can then balance user session traffic among the available servers, reducing user competition for connections on overworked 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 website 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 SLB 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 which is in high demand, it can become over-utilized. 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 under-utilized.

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

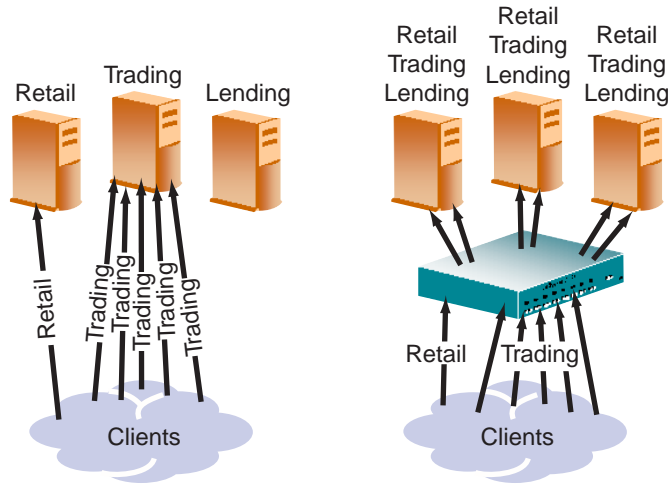


Figure 1-1 Traditional Versus SLB Network Configurations

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 WebSwitch, with SLB software, acts as a front-end to the servers, interpreting user session requests and distributing them among the available servers. To accomplish this, the switch is configured to act as a virtual server and given a virtual 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 real 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 real IP address of the best available real server and remaps the fields in each frame from virtual addresses to real addresses.

Network Topology Considerations

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

- In standard SLB, all client requests to a virtual IP address and all responses from the real servers must pass through the switch. If alternate paths exist between the client and the real servers (as shown in the figure below), the WebSwitch can be configured to proxy requests in order to guarantee that responses use the correct path (see [“Proxy IP Addresses for Complex SLB Networks”](#) on page 39).

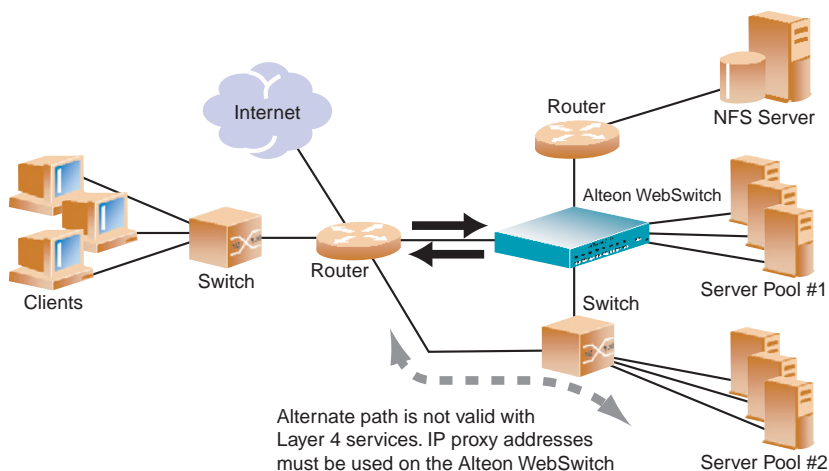


Figure 1-2 SLB Client/Server Traffic Routing

- 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 goes 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 8, “Persistence”](#)) or must use the minmisses or hash metrics (see [“Metrics for Real Server Groups”](#) on page 34).

- 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 configured to process client request traffic provide address translation from the virtual IP to the real server IP address.
 - Layer 4 server processing: ports configured to process server responses to client requests provide address translation from the real server IP address to the virtual IP address. These ports require real servers to be connected to the WebSwitch 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:

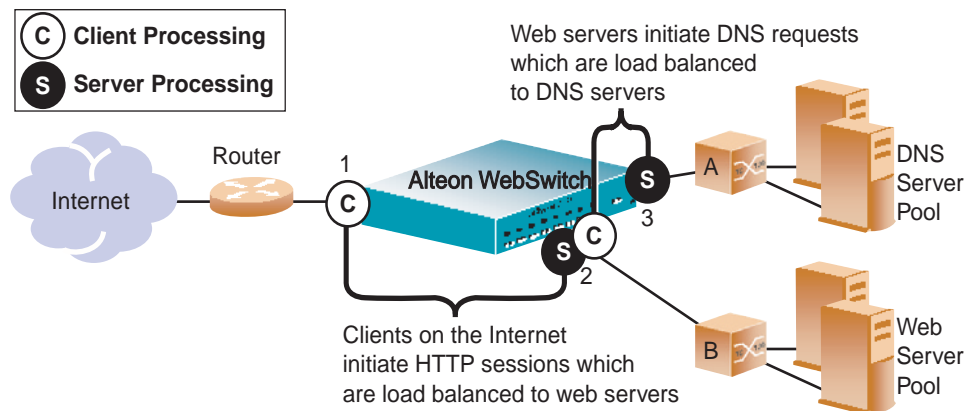


Figure 1-3 Example Network for Client/Server Port Configuration

In this figure, 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 is asked to perform both server and client processing.

Some topologies require special configuration. For example, if clients were added to switch B in the example above, these clients could not access the Web server pool using SLB services except through a proxy IP address configured on port 2 of the Alteon WebSwitch.

Virtual Matrix Architecture

Virtual Matrix Architecture (VMA) is a hybrid architecture that takes full advantage of the distributed processing capability in Alteon WebSwitches. 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).

The number of concurrent sessions per switch, with VMA enabled, is given below:

- AD4 and A184: 512K
- AD3 and A180E: 336K
- AD2: 256K

Configuration Issues

For better switch performance and higher session capacities, it is recommended that you enable VMA, especially when using Bandwidth Management and Content Intelligent Switching for multiple frames processing (up to 4,500 bytes).

Proxy IP Addresses and VMA

By default, VMA is enabled on the WebSwitch (`/cfg/slb/adv/matrix`). If you are upgrading to Web OS 8.3 from a previous release, however, VMA will be initially disabled if a proxy IP (PIP) address is configured for any port on the switch. VMA requires that if any port is configured with a PIP address, then all ports (except port 9) must be configured with a unique PIP address prior to enabling VMA.

With VMA, the concept of a per-port session table doesn't apply; instead, there is a global session table. To identify which processor should process responses to proxied requests, a unique PIP must be configured on each port (except port 9). The action of the unused PIPs can be disabled using `/cfg/slb/port x/proxy dis`.

Frames ingressing a port that has been configured with a PIP address and the `proxy` option enabled (`/cfg/slb/port x/proxy ena`) can be processed using a PIP address by any switch port, that is, the client source address will be substituted with the PIP on the port processing the request. Frames ingressing switch ports that have been configured with a PIP but

do not have the proxy option enabled can be processed by other ports configured with a PIP, but the client source address will not be replaced with a PIP address before being forwarded to a server.

```
>> # /cfg/slb/port 1/pip 10.10.10.10           (PIP used for NAT, etc.)
>> # /cfg/slb/port 1/pip 10.10.10.11/proxy ena (Turns on address proxying)
>> # /cfg/slb/port 2/pip 10.10.10.11/proxy dis (Turns off address proxying)
>> # /cfg/slb/port 3/pip 10.10.10.12/proxy dis
>> # /cfg/slb/port 4/pip 10.10.10.13/proxy dis
>> # /cfg/slb/port 5/pip 10.10.10.14/proxy dis
and so on....
```

FWLB and VMA

VMA must be enabled if you are setting up Firewall Load Balancing (FWLB) with clean-side switches performing SLB or URL-based SLB and DAM is enabled.

SLB Implementation

Web Hosting Configuration

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, so does the number of simultaneous Web connection requests.

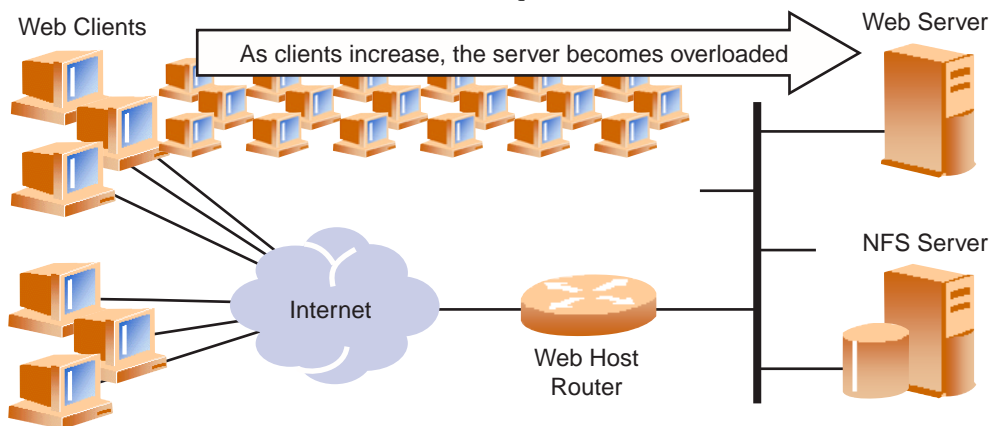


Figure 1-4 Web Hosting Configuration without SLB

Such a company has three primary needs:

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

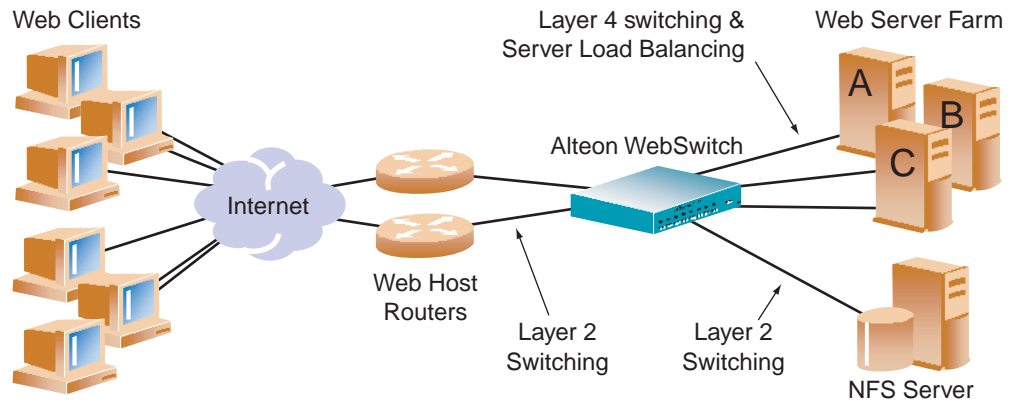


Figure 1-5 Web Hosting with SLB Solutions

Each concern about this company's site can be addressed by adding an Alteon WebSwitch with optional SLB software.

- Reliability is increased by providing multiple paths from the clients to the WebSwitch and access to a pool of servers that have 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.

Example Configuration for the Web Hosting Solution

In the following examples, many of the SLB options are left to their default values. See [“Additional SLB Options” on page 33](#) for more options.

The following is required prior to configuration:

- You must be connected to the switch command line interface as the administrator (see your *Web OS 8.3 Command Reference*).
- The optional SLB software must be enabled (see your *Web OS 8.3 Command Reference*).

NOTE – For details about any of the menu commands described in this example, see your *Web OS 8.3 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 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 Considerations” on page 24](#).

For this example, the three Web-host real servers have been given the following IP addresses on the same IP subnet:

Table 1-1 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 33](#)).

2. Define an IP interface on the switch.

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

3. On the switch, 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 # /cfg/slb/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 # /cfg/slb/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. On the switch, 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. On the switch, define a virtual server.

All client requests will be addressed to a virtual IP address on a virtual server defined on the switch. Clients acquire the virtual IP address through normal DNS resolution. In this example, HTTP is configured as the only service running on the 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 the command option information in your *Web OS 8.3 Command Reference*.

6. On the switch, define the port settings.

In this example, the following ports are being used on the WebSwitch:

Table 1-2 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. All three real servers get their Web content from this NFS server. This port does not require Web switching features.	None
5	Client router A connects the switch to the Internet where client requests originate.	Client
6	Client router B also 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# / cfg/slb/port 2	(Select physical switch port 2)
>> SLB port 2# server ene	(Enable server processing on port 2)
>> SLB port 2# / cfg/slb/port 3	(Select physical switch port 3)
>> SLB port 3# server ena	(Enable server processing on port 3)
>> SLB port 3# / cfg/slb/port 5	(Select physical switch port 5)
>> SLB port 5# client ena	(Enable client processing on port 5)
>> SLB port 5# / cfg/slb/port 6	(Select physical switch port 6)
>> SLB port 6# client ena	(Enable client processing on port 6)

7. On the switch, enable, apply, and verify the configuration.

>> SLB port 6# ..	(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 any appropriate changes.

8. On the switch, save your new configuration changes.

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

9. On the switch, check the Server Load Balancing information.

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

If necessary, make any appropriate configuration changes and then check the information again.

Additional SLB Options

In the examples above, many of the SLB options are left to their default values. The following configuration options can be used to tune the system.

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

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 IP address `172.16.10.45`, the unmasked portion of the virtual IP address (`0.0.0.45`) gets mapped directly to whichever real 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 UDP services, the switch pings servers to determine their status.

By default, the 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. By default, if a service does not respond to four consecutive health checks, the switch declares the service unavailable. Both the health check interval and the number of retries can be changed:

<code>>> # /cfg/slb/real <real server number></code>	<i>(Select the real server)</i>
<code>>> Real server# inter 4</code>	<i>(Check real server every 4 seconds)</i>
<code>>> Real server# retry 6</code>	<i>(If 6 consecutive health checks fail, declare real server down)</i>

More complex health-checking strategies may also be used. See [Chapter 10, “Health Checking”](#) for more details.

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 are `minmisses` (minimum misses), `hash`, `leastconns` (least connections), and `roundrobin`, explained in detail below and on the next page.

The default metric is `leastconns`. To change a real server group metric to `minmisses`, for example, enter:

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

Minimum Misses

The `minmisses` metric is optimized for Application Redirection. It uses IP address information in the client request to select a server. 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 option 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 address are 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 on the server between sessions. Server load with this metric becomes most evenly balanced as the number of active clients with different source or destination addresses increases.

When selecting a server, the switch will calculate a score for each available real server based on the relevant IP address information. The server that scores the highest 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.

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. 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, 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. This helps 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 leaves the mix, 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.

The hash metric provides more even 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 because of their ability to accept, process, and shut down connections more quickly than slower servers.

Round Robin

With the `roundrobin` metric, new connections are issued to each server in turn: the first real server in this 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.

Weights for Real Servers

Weights can be assigned to each real server. These weights bias load balancing to give the fastest real servers a larger share of connections during load balancing. Weight is specified as a number from 1 to 48. Each increment increases the number of connections the real server receives. 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:

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

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 released from the switch. To change the time-out period, enter the following:

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

The example above would change the time-out period of all connections on the designated real server to 4 minutes.

Maximum Connections for Real Servers

You can set the number of open connections each real server is allowed to handle for Server Load Balancing. 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 1,500 for quicker, multi-processor servers. The appropriate value depends on the duration of each session and how much CPU capacity is occupied by processing each session. Connections that using 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 handle overflow traffic when the maximum connection limit is reached. Each backup real server must be assigned a real server number, a real IP address, and then enabled. Finally, the backup must be assigned to each real server it will backup. 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 group #2 as backup)
```

Mapping Virtual Ports to Real Ports

Mapping to Single Ports

In addition to providing direct real server access in some situations (see [“Port Mapping” on page 43](#)), mapping is required when administrators choose to execute their real server processes on different TCP/UDP 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 – Port mapping is supported with Direct Access Mode (DAM) when filtering is enabled, a proxy IP address is configured, or URL parsing is enabled on any switch port. For information about DAM, refer to [“Direct Access Mode” on page 42](#).

Direct Server Return

Using the Direct Server Return (DSR) feature, the server can respond directly to the client, bypassing the switch. This capability is useful for sites where large amounts of data are going from servers to clients (such as 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 must go back through the switch for connection splicing.

NOTE – DSR requires that the server be set up to receive frames that have an IP destination address that is equal to the virtual server IP address(es).

The sequence of steps that are executed in this scenario are listed and pictured below:

1. A client request is forwarded to the WebSwitch.
2. Since 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, using the virtual IP address as the IP source address.

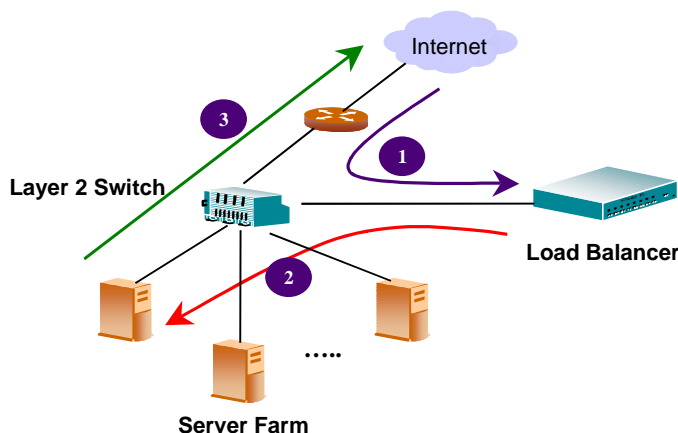


Figure 1-6 Direct Server Return

To set up direct server return, use the following commands:

```
>> # /cfg/slb/real <real server number>/submac ena
>> # /cfg/slb/virt <virtual server number>/service <service number>/nonat ena
```

Proxy IP Addresses for Complex SLB Networks

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 WebSwitch.

In complex network topologies, routers and other devices can create alternate paths around the WebSwitch managing SLB functions (see [Figure 1-2 on page 24](#)). Under such conditions, the client switch ports can use a proxy IP address.

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 PIP address is configured for the client port on the switch, the switch replaces the client's source IP address with the switch's own PIP 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 PIP address as the destination address for any response. This forces the SLB traffic 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 – Since requests appear to come from the switch PIP address rather than the client source IP address, use of proxy addresses can generate misleading information for network statistics or debugging.

The PIP address can also be used for direct access to the real servers (see [“Direct Client Access to Real Servers” on page 42](#)).

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. Reexamining the [“Example Configuration for the Web Hosting Solution” on page 29](#), the following revised port conditions are used:

Table 1-3 Proxy Example: Alteon 180 Port Usage

Port	Host	L4 Processing
1	Server A	None
2	Server B	None
3	Server C	None
4	Back-end NFS server. All three real servers get their Web content from the NFS server. This port does not require Web switching.	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# /cfg/slb/port 2      (Select switch port #2)
>> SLB port 2# server dis           (Disable server processing on port #2)
>> SLB port 2# /cfg/slb/port 3      (Select switch port #3)
>> SLB port 3# server dis           (Disable server processing on port #3)
```

2. Add proxy IP addresses to the client ports.

Each “client” port requires a PIP address. Each PIP address must be unique on your network. The following commands are used to configure client proxies:

```
>> # /cfg/slb/port 5                (Select network port #5)
>> SLB port 5# pip 200.200.200.68   (Set proxy IP address for client port #5)
>> SLB port 5# /cfg/slb/port 6      (Select network port #6)
>> SLB port 6# pip 200.200.200.69   (Set proxy IP address for client port #6)
```

The proxies are transparent to the user.

3. If the VMA feature is enabled, add PIP addresses for all other switch ports (except port 9).

VMA is normally enabled on the switch. In addition to enhanced resource management, this feature eliminates many of the restrictions found in earlier versions of the Web OS. It does require, however that when any switch port is configured with a PIP address, all ports must be configured with unique PIP addresses. Otherwise, if VMA is disabled, only the client port needs a PIP address and this step can be skipped.

The following commands can be used for configuring the additional unique PIP addresses:

```
>> SLB port 6# /cfg/slb/port 1           (Select network port 1)
>> SLB port 1# pip 200.200.200.70       (Set proxy IP address for port 1)
>> SLB port 1# /cfg/slb/port 2           (Select network port #2)
>> SLB port 2# pip 200.200.200.71       (Set proxy IP address for port #)
>> SLB port 2# /cfg/slb/port 3           (Select network port #3)
>> SLB port 3# pip 200.200.200.72       (Set proxy IP address for port 3)
>> SLB port 3# /cfg/slb/port 4           (Select network port #4)
>> SLB port 4# pip 200.200.200.73       (Set proxy IP address for port 4)
>> SLB port 4# /cfg/slb/port 7           (Select network port #7)
>> SLB port 7# pip 200.200.200.74       (Set proxy IP address for port 7)
>> SLB port 7# /cfg/slb/port 8           (Select network port 8)
>> SLB port 8# pip 200.200.200.75       (Set proxy IP address for port 8)
```

NOTE – Port 9 does not require a PIP address under VMA.

See “[Virtual Matrix Architecture](#)” on page 26 and the *Web OS 8.3 Command Reference* for more information (/cfg/slb/adv/matrix).

4. Apply and save your changes.

NOTE – Remember that you must apply any changes in order for them to take effect and save changes if you wish them to remain in effect after switch reboot. Also, the /info/slb command is useful for checking the state of SLB operations.

Direct Client Access to Real Servers

Some clients may need direct access to the real servers (for example, to monitor a real server from a management workstation). This access can be provided in a number of ways:

- Direct Access Mode (DAM)
- Multiple IP addresses on the server
- PIP addresses
- Port mapping
- Management network

Direct Access Mode

When DAM (`/cfg/slb/direct`) is enabled on a switch, any client can communicate with any real server's load-balanced service. Also, in DAM, any number of 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 PIP address is configured, or URL parsing is enabled on any switch port.

Multiple IP Addresses on the Server

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.

Proxy IP Addresses

PIP addresses are used primarily to eliminate SLB topology restrictions in complex networks (see [“Network Topology Considerations” on page 24](#)). PIP addresses can also provide direct access to real servers.

If the switch port to the client is configured with a PIP address (see [“Proxy IP Addresses for Complex SLB Networks” on page 39](#)), the client can access each real server directly using the real server's IP address. This requires that the switch port connected to the real server has server and client processing disabled (see the `server` and `client` options under `/cfg/slb/port` in the *Web OS 8.3 Command Reference*).

SLB is still accessed using the virtual server IP address.

Port Mapping

When SLB is used without PIP addresses and without DAM, the 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 switch, with the real server IP address getting remapped back to the virtual server IP address.

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 must be mapped to the proper real server port.

In [Figure 1-7](#), clients can access SLB services through well-known TCP port 80 at the virtual server's IP address. This 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 at the real server's IP address.

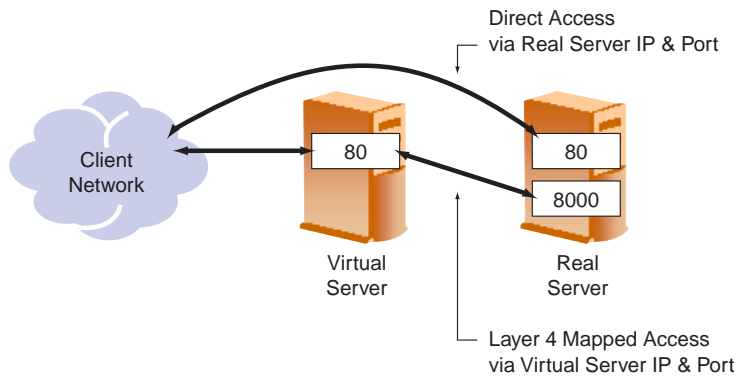


Figure 1-7 Mapped and Non-Mapped Server Access

NOTE – Port mapping is supported with DAM when filtering is enabled, a PIP address is configured, or URL parsing is enabled on any switch port.

Management Network

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`), 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 services being load balanced.

The `mnet` and `mmask` options are described below:

- `mnet`: If defined, management traffic with this source IP address will be 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.

FTP Server Load Balancing

Web OS 8.3 supports load balancing of File Transfer Protocol (FTP) servers on both public and private networks for both active and passive FTP modes. To do this, the switch modifies FTP commands from both the client (for active FTP) and server (for passive FTP).

NOTE – This feature does not support different FTP modes within a single session - that is, the user cannot switch from active to passive or vice versa in the same FTP session.

- For passive FTP SLB, the switch watches for the `PASV` command from the FTP client and modifies the “entering passive mode” command coming back from the FTP server. It replaces the real IP (RIP) address with a virtual IP (VIP) address and the real server port (RPORT) with a virtual port (VPORT) so that the client will make an active open connection between the client data port and the switch, instead of the server. The switch then re-maps requests to the FTP server to which the control channel was bound.
- For active FTP SLB, the switch watches for the `PORT` command from the FTP client and translates the FTP server active open connection to use *VIP:VPORT* instead of *RIP:RPORT*. The server data connection will look as if it came from the switch itself.

Background

As defined in RFC 959, FTP uses two channels/connections—one for control information and another for data. Each connection is unique. Unless the client requests a change, the server is always using 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. Since 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.

Configuring FTP SLB

NOTE – You must use either DAM or a PIP address.

1. Make sure a PIP address is enabled on the client port(s), or DAM is enabled.
2. Make sure the virtual port for FTP is already set up for the virtual server.
3. Enable FTP parsing, using this command:

```
>> # /cfg/slb/virt <virtual server number>/ftpp ena
```

4. To make your configuration changes active, enter **apply** at any prompt in the CLI.

```
>> # /cfg/slb/virt <virtual server number>/apply
```

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

Mapping a Single Virtual Port to Multiple Real Ports

To take advantage of multi-CPU or multi-process servers, Web OS 8.3 enables the network administrator to map a single virtual port to multiple real ports. This capability allows the Website managers, for example, to differentiate users of a service by using multiple service ports to process client requests.

An Alteon WebSystems' switch running Web OS 8.3 supports up to eight real ports per server, when multiple rports is enabled. This feature enables the network administrator to configure up to eight remote ports for a single service port on all of the Alteon 180 series and AD series platforms. This feature is supported in Layer 4 and Layer 7 and in cookie-based and SSL persistence switching environments.

NOTE – For each real server, you can only configure one service with multiple real ports.

Health Checking Issues

- If multi-rport is enabled, each server's service will be added to the service's health-check table.
- If any service running on a real server fails, that server is removed from the real server group, and the server will be taken offline. Traffic can only be directed to a particular server in a real server group if *all* of the services configured on that server are up and available.

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.

Mapping a Virtual Port to Multiple Real Ports

To support this feature, the switch will use the IP address and port number combination in making all the load-balancing decisions.

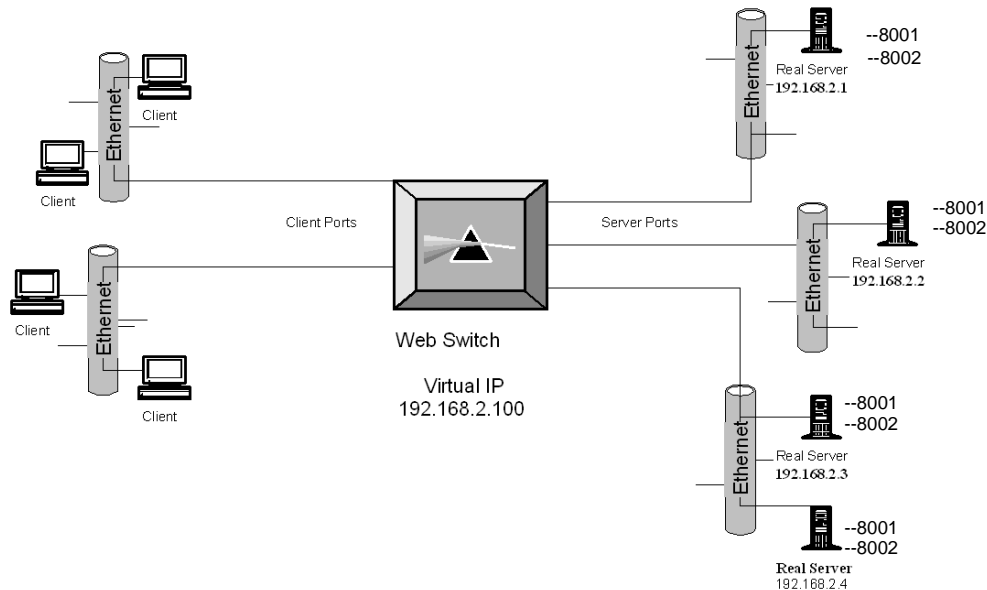


Figure 1-8 Basic Virtual Port to Real Port Mapping Configuration

Domain Name	Virtual 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 configuration, the same real servers are used to support a single service. The service is 192.168.2.100/80 (virtual server IP address/virtual port). The real servers in this example are: 192.168.2.1, 192.168.2.2, 198.168.2.3, and 198.168.2.4.

With a switch running Web OS 8.3, multiple real ports can be mapped to a virtual port. Internally, the switch treats the real server IP address/port mapping combination as a real server. In this example, the logical real servers are:

- 192.168.2.1/8001(RIP 1/rport 1)
- 192.168.2.1/8002(RIP 1/rport 2)
- 192.168.2.2/8001(RIP 2/rport 1)
- 192.168.2.2/8002(RIP 2/rport 2)
- 192.168.2.3/8001(RIP 3/rport 1)
- 192.168.2.3/8002(RIP 3/rport 2)
- 192.168.2.4/8001(RIP 4/rport 1)
- 192.168.2.4/8002(RIP 4/rport 2)

If the load-balancing algorithm being used is round robin, then the switch sends the incoming connections to these eight logical real servers in a round robin fashion.

If the algorithm is least connections, the switch sends the incoming connections to the RIP/rport combination with the least number of connections.

Using the /cfg/slb/virt Command

This command defines the real server TCP or UDP port assigned to this 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/service <virtual port>, the switch will map the virtual port to this 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.

Using the /cfg/slb/real Command

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`.

Configuration Example for Multiple Rport:

1. Configure a Virtual IP address.

```
>> # /cfg/slb/virt 1/vip 192.168.2.100
```

2. Configure four real servers.

```
>> # /cfg/slb/real 1/rip 192.168.2.1
>> # ../real 2/rip 192.168.2.2
>> # ../real 3/rip 192.168.2.3
>> # ../real 4/rip 192.168.2.4
```

3. Add all four servers to a group.

```
>> # /cfg/slb/group 1
```

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)



CHAPTER 2

Filtering

This chapter provides a conceptual overview of filters and configuration examples showing how filters can be used to ensure network security and redirect traffic.

Overview

Alteon WebSwitches can be utilized to not only deliver content efficiently, but also to secure your servers from unauthorized intrusion, probing, and Denial-Of-Service (DOS) attacks. The capabilities built into Web OS 8.3 allow for extensive filtering at the IP and TCP/UDP levels.

Benefits

Layer 3 (IP) and Layer 4 (application) filtering gives the network administrator a powerful tool with the following benefits:

- Filtering increases security for server networks.

Filters can be configured to allow or deny traffic according to various IP address, protocol, and Layer 4 port criteria. This gives the administrator control over the types of traffic permitted through the switch. Any filter can optionally be configured to generate `syslog` messages for increased security visibility.

- Generic Network Address Translation (NAT)

NAT can be used to map the source or destination IP address and port of private network traffic to/from an advertised network IP address and port.

Filtering Criteria

Up to 224 filters can be configured on the switch. Each filter can be set to allow, deny, redirect, or translate traffic based on any combination of the following criteria:

- Source IP Address or range
- Destination IP Address or range
- Protocol type (for example: IP, UDP, TCP, ICMP, and others)
- TCP flags
- Application, source port, or range (For example: ftp, http, telnet, 31000-33000, etc.)
- Application, destination port, or range (For example: ftp, http, telnet, 31000-33000, etc.)
- Inverse: activate the filter whenever the specified conditions are *not* met.

For example, you can 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.

Below are a list of the well-known protocols and applications.

Table 2-1 Well-Known Protocol Types

Number	Protocol Name
1	icmp
2	igmp
6	tcp
17	udp
89	ospf
112	vrrp

Table 2-2 Well-Known Application Ports

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

Stacking Filters

Once configured, 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 224. 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 doesn’t 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:

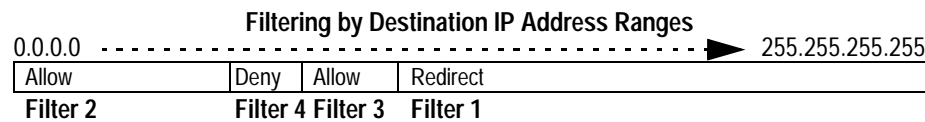


Figure 2-1 Assigning Filters According to Range of Coverage

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:

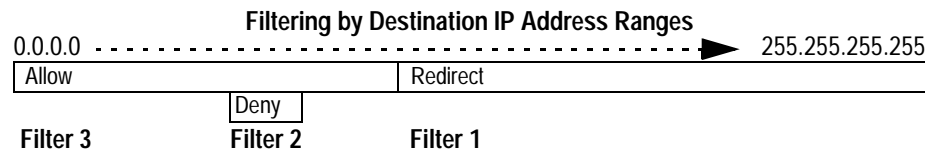


Figure 2-2 Assigning Filters to Overlapping Ranges

In this example, the “deny” filter must be processed prior to the “allow” filter. If the “allow” filter was permitted to take precedence, the “deny” filter 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:

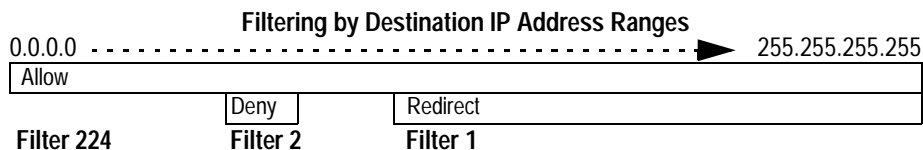


Figure 2-3 Assigning a Default Filter

In this example, the default filter is defined as filter 224 in order to give it the lowest order of precedence. All matching criteria in filter 224 is set to the “any” state. If no other filter acts on the traffic, filter 224 handles it, denying and logging unwanted traffic.

```
>> # /cfg/slb/filt 224                (Select the default filter)
>> Filter 224# sip any                (From any source IP addresses)
>> Filter 224# dip any                (To any destination IP addresses)
>> Filter 224# proto any              (For any protocols)
>> Filter 224# action deny            (Deny matching traffic)
>> Filter 224# ena                    (Enable the default filter)
>> Filter 224# adv/log enable         (Log all matching traffic to syslog)
```

Although recommended when configuring filters for IP traffic control and redirection, default filters are not required. 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.

Numbering Filters

You may wish to consider numbering your filters by increments of 5 or 10 (for example: 5, 10, 15, 20, etc.) so that filters could be easily inserted between others in the list, if required.

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 to create and apply the following filter:

>> # /cfg/slb/filt 15	<i>(Select filter 15)</i>
>> Filter 15# sip any	<i>(From any source IP address)</i>
>> Filter 15# dip any	<i>(To any destination IP address)</i>
>> Filter 15# action allow	<i>(Allows matching traffic to pass)</i>
>> Filter 15# proto icmp	<i>(For the ICMP protocol)</i>
>> Filter 15# ena	<i>(Enable the filter)</i>
>> Filter 15# adv/log enable	<i>(Create a log entry when matched)</i>
>> Filter 15# /cfg/slb/port 7	<i>(Select a switch port to filter)</i>
>> SLB port 7# add 15	<i>(Add the filter to the switch port)</i>
>> SLB port 7# filt ena	<i>(Enable filtering on the switch port)</i>
>> SLB port 7# apply	<i>(Apply the configuration changes)</i>
>> SLB port 7# save	<i>(Save the configuration changes)</i>

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
--

Security Example

Consider the following sample network:

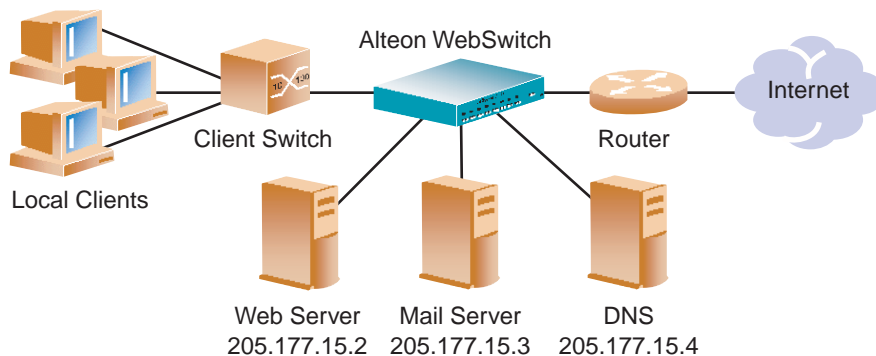


Figure 2-4 Example Security Topology

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.

For best security, it is commonly considered that you should configure filters to deny all traffic except for those services you specifically wish to allow. 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 will be denied and logged.

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.

Example Configuration for the Security Solution

Prior to configuration, you must be connected to the switch command line interface as the administrator.

In this example, all filters will be applied only to the switch port that connects to the Internet. If intranet restrictions were required, filters could 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 the *Web OS 8.3 Command Reference* for a list of other well-known protocols and services.

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 2-3 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. On the switch, create a default filter that will deny and log unwanted traffic.

The default filter is defined as filter 224 in order to give it the lowest order of precedence:

>> # /cfg/slb/filt 224	(Select the default filter)
>> Filter 224# sip any	(From any source IP addresses)
>> Filter 224# dip any	(To any destination IP addresses)
>> Filter 224# proto any	(For any protocols)
>> Filter 224# action deny	(Deny matching traffic)
>> Filter 224# ena	(Enable the default filter)
>> Filter 224# adv/log enable	(Log matching traffic to syslog)

NOTE – When the `proto` parameter is *not* `tcp` or `udp`, then `sport` and `dport` are ignored.

3. On the switch, 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 224# ../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    (Fill 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# ena                      (Enable the filter)
```

4. On the switch, 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    (Fill 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    (Fill 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. On the switch, 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 originating 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# ena                     (Enable the filter)
```

6. On the switch, 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. On the switch, 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                (Select the menu for Filter #6)
>> Filter 6# sip any                  (From any source IP address)
>> Filter 6# dip 205.177.15.4         (To local DNS Server)
>> Filter 6# dmask 255.255.255.255   (Fill mask for exact dest. address)
>> Filter 6# proto udp                (For UDP protocol traffic)
>> Filter 6# sport any                (From any source port)
>> Filter 6# dport domain             (To any DNS destination port)
>> Filter 6# action allow             (Allow matching traffic to pass)
>> Filter 6# ena                     (Enable the filter)
>> Filter 6# ../filt 7                (Select the menu for Filter #7)
>> Filter 7# sip 205.177.15.4         (From local DNS Server)
>> Filter 7# smask 255.255.255.255   (Fill mask for exact source address)
>> Filter 7# dip any                  (To any destination IP address)
>> Filter 7# proto udp                (For UDP protocol traffic)
>> Filter 7# sport domain             (From a DNS source port)
>> Filter 7# dport any                (To any destination port)
>> Filter 7# action allow             (Allow matching traffic to pass)
>> Filter 7# ena                     (Enable the filter)
```

Similarly, for TCP:

```
>> Filter 7# ../filt 8                (Select the menu for Filter #8)
>> Filter 8# sip any                  (From any source IP address)
>> Filter 8# dip 205.177.15.4         (To local DNS Server)
>> Filter 8# dmask 255.255.255.255   (Fill mask for exact dest. address)
>> Filter 8# proto tcp                (For TCP protocol traffic)
>> Filter 8# sport any                (From any source port)
>> Filter 8# dport domain             (To any DNS destination port)
>> Filter 8# action allow             (Allow matching traffic to pass)
>> Filter 8# ena                     (Enable the filter)
>> Filter 8# ../filt 9                (Select the menu for Filter #9)
>> Filter 9# sip 205.177.15.4         (From local DNS Server)
>> Filter 9# smask 255.255.255.255   (Fill mask for exact source address)
>> Filter 9# dip any                  (To any destination IP address)
>> Filter 9# proto tcp                (For TCP protocol traffic)
>> Filter 9# sport domain             (From a DNS source port)
>> Filter 9# dport any                (To any destination port)
>> Filter 9# action allow             (Allow matching traffic to pass)
>> Filter 9# ena                     (Enable the filter)
```

8. On the switch, assign the filters to the switch port that connects to the Internet.

```
>> Filter 9# ../port 5                (Select the SLB port 5 to the Internet)
>> SLB Port 5 # add 1                  (Add filter 1 to port 5)
>> SLB Port 5 # add 2                  (Add filter 2 to port 5)
>> SLB Port 5 # add 3                  (Add filter 3 to port 5)
>> SLB Port 5 # add 4                  (Add filter 4 to port 5)
>> SLB Port 5 # add 5                  (Add filter 5 to port 5)
>> SLB Port 5 # add 6                  (Add filter 6 to port 5)
>> SLB Port 5 # add 7                  (Add filter 7 to port 5)
>> SLB Port 5 # add 8                  (Add filter 8 to port 5)
>> SLB Port 5 # add 9                  (Add filter 9 to port 5)
>> SLB Port 5 # add 224                (Add the default filter to port 5)
>> SLB Port 5 # filt enable            (Enable filtering for port 5)
```

9. On the switch, apply and verify the configuration.

```
>> SLB Port 5 # apply                  (Make your changes active)
>> SLB Port 5 # cur                    (View current settings)
```

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

10. On the switch, save your new configuration changes.

```
>> SLB Port 5 # save                  (Save for restore after reboot)
```

11. On the switch, check the Server Load Balancing information.

```
>> SLB Port 5 # /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.

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 *Web OS 8.3 Command Reference*).

TCP ACK Matching for Filters

The ACK filter criteria provides greater filtering flexibility. When the ack option is enabled, the filter matches only those frames set with the TCP ACK or RST flag. The ACK criteria appears in the Web OS Web interface and in the command line interface on the Filter Menu (`/cfg/slb/filt <filter-number>/adv`).

Example: Consider the following network:

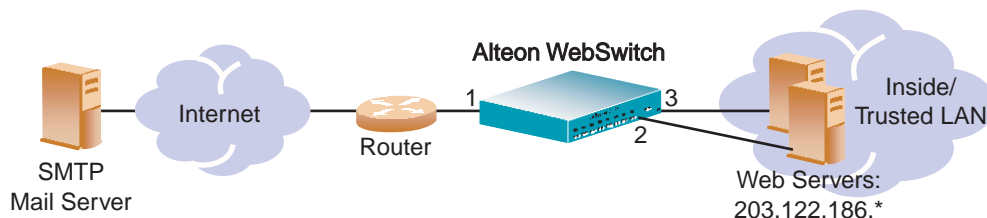


Figure 2-5 Example Filter TCP ACK Matching Network

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 destination TCP port 25, and the SMTP server will acknowledge each TCP session and data transfer using source TCP port 25.

Filtering with the ACK flag closes one potential security hole. Without it, the switch would permit a TCP SYN connection request to reach any listening destination TCP 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.

This filter with the ACK flag requirement prevents external servers from beginning a TCP connection (with a TCP SYN) from source TCP port 25. The server will drop any frames that have the ACK flag turned off in them.

The following filters are required:

1. One filter must allow 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. One filter must allow 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) 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# ack ena	<i>(For acknowledgments only)</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>

3. One filter must allow trusted HTTP traffic from the Internet to pass through the switch to the Web servers.

>> Filter 15# ../filt 16	<i>(Select a filter for incoming HTTP traffic)</i>
>> Filter 16# sip any	<i>(From any source IP address)</i>
>> Filter 16# sport http	<i>(From well-known source HTTP 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 15# dport http	<i>(To well-known destination HTTP port)</i>
>> Filter 16# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 16# ena	<i>(Enable the filter)</i>

4. One filter must allow HTTP responses from the Web servers to pass through the switch to the Internet.

>> Filter 16# ../filt 17	<i>(Select a filter for outgoing HTTP traffic)</i>
>> Filter 17# sip 203.122.186.0	<i>(From the Web servers' source IP address)</i>
>> Filter 17# smask 255.255.255.0	<i>(From the entire subnet range)</i>
>> Filter 17# sport http	<i>(From well-known source HTTP port)</i>
>> Filter 17# proto tcp	<i>(For TCP traffic)</i>
>> Filter 17# dip any	<i>(To any destination IP address)</i>
>> Filter 17# dport http	<i>(To well-known destination HTTP port)</i>
>> Filter 17# action allow	<i>(Allow matching traffic to pass)</i>
>> Filter 17# ena	<i>(Enable the filter)</i>

5. One default filter is required to deny everything else.

>> Filter 17# ../filt 224	<i>(Select a default filter)</i>
>> Filter 220# sip any	<i>(From any source IP address)</i>
>> Filter 220# dip any	<i>(To any destination IP address)</i>
>> Filter 220# action deny	<i>(Block matching traffic)</i>
>> Filter 220# ena	<i>(Enable the filter)</i>

6. Next, the filters must be applied to the appropriate switch ports.

>> Filter 220# ../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 HTTPS filter)</i>
>> SLB port 1# add 224	<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 17	<i>(Add the outgoing HTTP filter to the port)</i>
>> SLB port 2# add 224	<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 17	<i>(Add the outgoing HTTP filter to the port)</i>
>> SLB port 3# add 224	<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>

Option to List ICMP Filtering Types

Web OS 8.3 provides an option to the ICMP command to list both available ICMP types and a usage help map listing the valid ICMP number/names that can be entered.

When a Command Line Interface (CLI) user is prompted to enter an ICMP type for filtering, the user often does not know the abbreviated names that are used on the switch. The Web OS 8.3 CLI includes a mechanism for showing users what ICMP types are available and the correct vocabulary for specifying the ICMP type on which to filter.

```
>> # /cfg/slb/filt/adv/icmp list
```

When the user types in the command shown above, an ICMP-type listing is displayed. You can see this option when you type the command `help icmp`.

```
>> Filter 1 Advanced# help icmp
Usage: icmp any|<number>|<type; icmp list for list>
```

```
>> Filter 1 Advanced# icmp list

ICMP types:

 0      echorep      echo reply
 3      destun       destination unreachable
 4      quench       source quench
 5      redir        redirect
 8      echoreq      echo request
 9      rtradv       router advertisement
10      rtrsol       router solicitation
11      timex        time exceeded
12      param        parameter problem
13      timereq      timestamp request
14      timerep      timestamp reply
15      inforeq      information request
16      inforep      information reply
17      maskreq      address mask request
18      maskrep      address mask reply
```

Network Address Translation Examples

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. Private networks can use whatever IP addresses they please, including those that are in use elsewhere on the Internet, or reserved for other purposes.

Private networks serve two main purposes: First, because private IP addresses are not valid or visible outside the private network, they can increase network security; second, since valid, registered IP addresses are a limited resource, many companies use private IP addresses to create internal networks much larger than could be created using only their official addresses.

With Network Address Translation (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.

Internal Client Access to Internet

In this dynamic NAT example, clients on the internal private network require TCP/UDP access to the Internet:

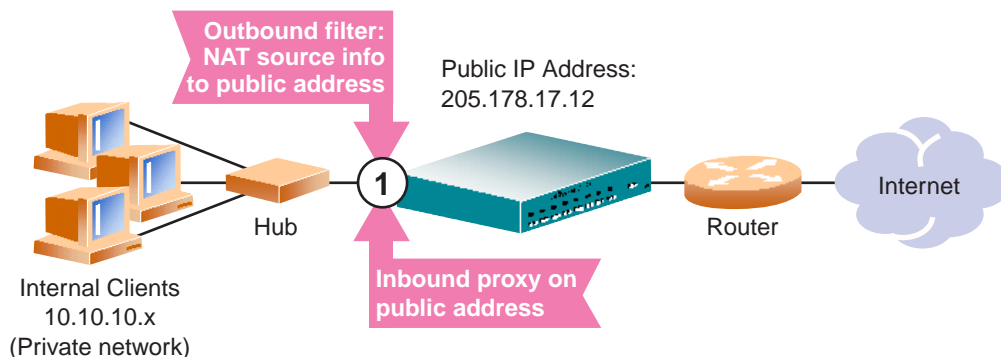


Figure 2-6 Dynamic NAT

This example requires a NAT filter to be configured on the switch port 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. In addition, the public IP address must be configured as a proxy IP address on the switch port connected to the internal clients. The proxy performs the reverse translation, restoring the private network addresses on inbound packets.

This 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.

This example could be configured as follows:

NOTE – The `invert` option is only specific to this example and is not required.

>> # /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 adv/proxy enable	(Allow PIP proxy translation)
>> Filter 14 ena	(Enable the filter)
>> Filter 14 ../port 1	(Select SLB port 1)
>> SLB port 1# add 14	(Add the filter to port 1)
>> SLB port 1# pip 205.178.17.12	(Set public IP address proxy)
>> 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)

NOTE – Dynamic NAT solutions apply only to TCP/UDP traffic. Also, filters for dynamic NAT should be placed behind any static NAT filters (see next example). Dynamic filters should be given higher filter numbers.

External Client Access to Server

In this example, clients on the external Internet require access to a server on the private network:

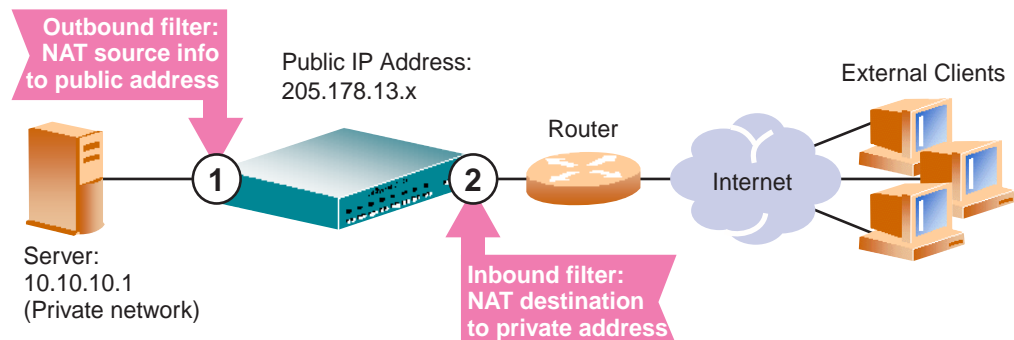


Figure 2-7 Static NAT

This 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.

This could be configured as follows:

>> # /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 PIP proxy 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 PIP proxy 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 be placed ahead of dynamic NAT filters (previous example). Static filters should be given lower filter numbers.

Defining IP Address Ranges for Filters

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 `sip` (source) or `dip` (destination) defines the base IP address in the desired range, and the `smask` (source) or `dmask` (destination) 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 `dip`.

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 2-4 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

Web OS 8.3 Filtering Additions

Web OS software supports filtering on all IP options, ICMP message types, and TCP flags.

In general, the switch ignores the presence of IP options when matching a frame to a filter. When configuring filters, observe the following:

- **IP Options:** Filtering on all options can either be enabled or disabled. Filtering a single option is not supported.
- **ICMP message types:** Only one message type can be set at any one time.
- **TCP Flags:** More than one TCP flag can be set at the same time. If there is more than one flag enabled, the flags are applied with a logical AND operator.

Example: By setting the switch to filter SYN and ACK, the switch will filter all SYN-ACK frames.

- These filtering options work only with cache-disabled filtering.

Exercise caution when applying cache-enabled and cache-disabled filters to the same switch port. A cache-enabled filter creates a session entry in the switch, so that the switch can bypass checking for subsequent frames that match the same criteria. This can potentially cause cache-disabled filters applying to the same switch port to be bypassed.

Full TCP Flag Filtering

Web OS 8.3 supports packet filtering based on any or all TCP flags.

NOTE – All TCP options are disabled in the default filter configuration. What this means is that packets with TCP flags will *not* be inspected unless one or more TCP options are enabled.

Table 2-5 TCP Flags

Flag	Description
URG	Urgent
ACK	Acknowledgement
PSH	Push
RST	Reset
SYN	Synchronize
FIN	Finish

Configuring the TCP Filter

1. Set up a filter as you would under a normal situation, configuring options such as sip, smask, dip, and smask, as needed.

```
>> # /cfg/slb/filt <filter number>/sip <IP address>
```

2. Go to the TCP Flags Advanced Menu.

```
>> # Filter 1# adv/tcp
```

3. Enable the flag you wish to be used to filter TCP SYN packets.

For example:

```
>> # TCP Flags Advanced Menu# syn e
```

4. Add the appropriate filter on the port that needs to be filtered.

```
>> # /cfg/slb/port <port number>/add <filter ID (1-224)>
```

5. Enable filtering on the port.

```
>> # filt ena
```

6. To make your configuration changes active, enter apply at any prompt in the CLI.

```
>> # apply
```

ICMP Type Filtering

Web OS 8.3 supports packet filtering based on any or all ICMP types.

NOTE – In general, the switch ignores the presence of ICMP options when matching a frame to a filter. Packets with ICMP message types will NOT be filtered out unless ICMP options are enabled.

Table 2-6 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

Configuring the ICMP Filter

NOTE – If you want to configure ICMP type filters, you should disable the `cache` option in the Filter Advanced Menu (`cfg/slb/filt <filter-number>/adv`).

1. Set up a filter as you would under a normal situation, configuring options such as `sip`, `smask`, `dip`, and `smask`, as needed.

```
>> # /cfg/slb/filt <filter number>/sip <IP address>
```

2. Set protocol type to ICMP.

```
>> # proto icmp
```

3. Go to the Filter Advanced Menu and select the ICMP option. Then, enter the option you want to be used to filter ICMP packets:

```
>> # adv/icmp
Current ICMP message type:      any
Enter ICMP message type or any: <message-type/number>
```

4. Add the appropriate filter on the port that needs to be filtered.

```
>> # /cfg/slb/port <port number>/add <filter ID (1-224)>
```

5. Enable filtering on the port.

```
>> # filt ena
```

6. To make your configuration changes active, enter `apply` at any prompt in the Command Line Interface (CLI).

FTP Client NAT (Active FTP for Dynamic NAT)

Alteon WebSystems switches provide Network Address Translation (NAT) services to many clients with private IP addresses. However, on switches running Web OS 6.0, clients using active FTP cannot send a request to a remote FTP server when their client IP address is private. In Web OS 8.3, an FTP enhancement now provides the capability to perform true FTP NAT for dynamic NAT.

Because of the way FTP works in active mode, a client will send 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.

In Web OS, the switch can monitor the control channel and replace the client's private IP address with a proxy IP (PIP) 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 IP address will be replaced by a public proxy IP address, using a pool of proxy IP addresses instead of a single one.
- The real server port will be replaced with a proxy port.

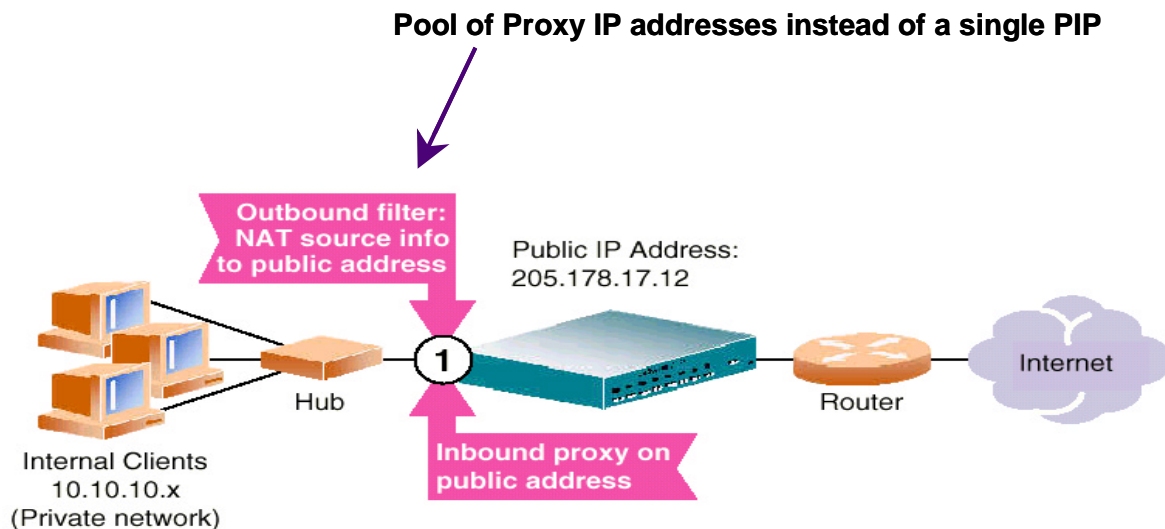


Figure 2-8 Many to Many NAT

Configuring Active FTP Client NAT

NOTE – The passive mode does not need this feature.

1. Make sure there's a proxy IP address enabled on the filter port.
2. Make sure there's a source NAT filter set up for the port.
3. Enable active FTP NAT using the following command:

```
>> # /cfg/slb/filt <filter number>/adv/ftpa e
```

4. Apply and save the switch configuration.



CHAPTER 3

Application Redirection

Application Redirection improves network bandwidth and provides unique network solutions. Filters can be created to redirect traffic to cache and application servers. Repeated client access to common Web or application content across the Internet can be an inefficient use of network resources. By redirecting client requests to a local Web cache or application server, you increase the speed at which clients access the information and free-up valuable network bandwidth.

NOTE – To access Application Redirection functionality, the optional Layer 4 software must be enabled in the switch (see “Filtering and Layer 4” in Chapter 8 of the *Web OS 8.3 Command Reference*).

Overview

Much of the information downloaded from the Internet is not unique, as clients will often access the same information many times as they return to a Web page for additional information or to explore other links. Duplicate information also gets requested as the components that make up Internet data at a particular website (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.

Web cache redirection can help alleviate the congestion seen at your Internet router. When Application Redirection filters are properly configured for your Web OS-powered switch, outbound client requests for Internet data are intercepted and redirected to a group of Web cache servers on your network. The Web cache servers duplicate and store inbound Internet data that has been requested by your clients. If the Web cache servers recognize a client’s outbound request as one that can be filled with cached information, the Web cache servers will supply the information rather than sending the request out across the Internet.

In addition to increasing the efficiency of your network, access to locally cached information can be granted much faster than requesting the same information across the Internet.

Web Cache Redirection Environment

Consider a network where client HTTP requests begin to regularly overload the Internet router.

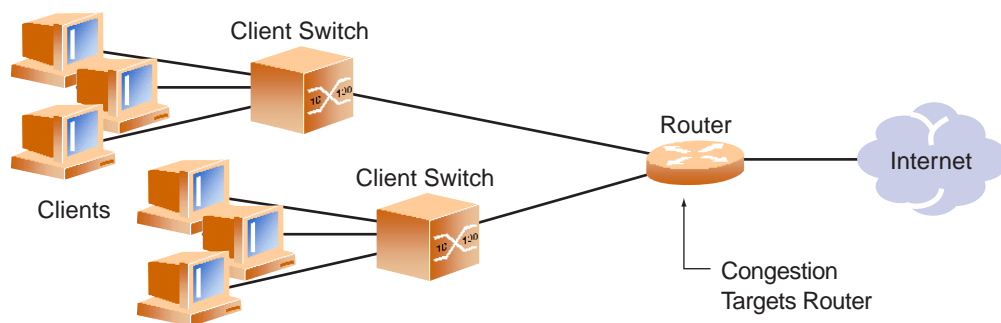


Figure 3-1 Traditional Network Without Web Cache Redirection

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.

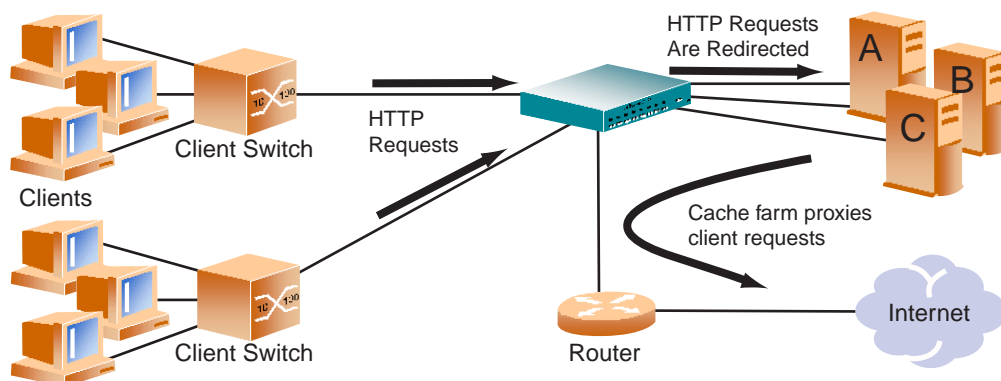


Figure 3-2 Network with Web Cache Redirection

Adding an Alteon WebSystems switch with optional Layer 4 software addresses these issues:

- Web cache servers can be added or removed dynamically without interrupting services.
- Performance is improved by balancing the cached Web 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.

Example Web-Cache Solution Configuration

The following is required prior to configuration:

- You must be connected to the switch command line interface as the administrator.
- Optional Layer 4 software must be enabled.

NOTE – For details about the procedures above, and about any of the menu commands described in this example, see the *Web OS 8.3 Command Reference*.

In this example, an Alteon WebSwitch is placed between the clients and the border gateway to the Internet. The switch will be configured to intercept all Internet bound HTTP requests (on default TCP port 80), and redirect them to the Web cache servers. The switch will distribute HTTP requests equally to the Web cache servers based on the destination IP address of the requests.

Also, filters are not limited to the few protocols and TCP or UDP applications shown in this example. See the *Web OS 8.3 Command Reference* for a list of other well-known protocols and services.

1. Assign an IP address to each of the Web cache servers.

Just as with SLB, the Web cache real servers will be 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 switch that will perform the Web cache redirection. In addition, the path from the switch to the real servers must not contain a router. The router would stop HTTP requests from reaching the Web cache servers, instead directing them back out to the Internet.

More complex network topologies can be used if configuring IP proxy addresses (see “[IP Proxy Addresses](#)” on page 84).

For this example, the three Web cache real servers have the following IP addresses on the same IP subnet:

Table 3-1 Web Cache Example: Real Server IP addresses

Web 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 Web cache software on all three Web cache servers.
3. Define an IP interface on the switch.

Because, by default, the switch only re-maps destination MAC addresses, it must have an IP interface on the same subnet as the three Web cache servers.

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. On the switch, define each real server.

For each Web 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. On the switch, define a real server group.

This places the three Web 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. On the switch, set the real server group metric to `minmisses`.

This helps minimize Web cache misses in the event real servers fail or are taken out of service:

```
>> Real server group 1 # metrc minmisses (Metric for minimum cache misses.)
```

7. On the switch, 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 SLB. To disable server processing on the port, use the commands on the `/cfg/slb/port` menu, as described in Chapter 8 of the *Web OS 8.3 Command Reference*.

8. On the switch, 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` 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 transparent proxies are used for NAT on the switch (see [Step 3. on page 80](#)), 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 switch, make sure to use the service port required by the specific software package.

See [“IP Proxy Addresses” on page 84](#) for more about IP proxy addresses.

9. On the switch, create a default filter.

In this case, the default filter will allow all non-cached traffic to proceed normally:

```
>> Filter 2# ../filt 224                (Select the default filter)
>> Filter 224# sip any                  (From any source IP addresses)
>> Filter 224# dip any                  (To any destination IP addresses)
>> Filter 224# proto any                (For any protocols)
>> Filter 224# action allow             (Set the action to allow traffic)
>> Filter 224# ena                     (Enable the default filter)
```

NOTE – When the `proto` parameter is not `tcp` or `udp`, then `sport` and `dport` are ignored.

10. On the switch, 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 224# ../port 5                (Select the SLB port 5)
>> SLB Port 5 # add 2                   (Add filter 1 to port 5)
>> SLB Port 5 # add 224                 (Add the default filter to port 5)
>> SLB Port 5 # filt enable             (Enable filtering for port 5)
>> SLB Port 5 # ../port 6              (Select the SLB port 6)
>> SLB Port 6 # add 2                   (Add filter 1 to port 6)
>> SLB Port 6 # add 224                 (Add the default filter to port 6)
>> SLB Port 6 # filt enable             (Enable filtering for port 6)
```

11. On the switch, enable, apply, and verify the configuration.

```
>> SLB Port 6 # /cfg/slb                (Select Server Load Balancing Menu)
>> Layer 4# on                          (Activate Layer 4 software services)
>> Layer 4# apply                        (Make your changes active)
>> Layer 4# cur                          (View current settings)
```

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 switch (see “Activating Optional Software” in the *Web OS 8.3 Command Reference*).

12. Examine the resulting information from the “cur” command. If any settings are incorrect, make appropriate changes.

13. On the switch, save your new configuration changes.

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

14. On the switch, 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 *Web OS 8.3 Command Reference*).

IP Proxy Addresses

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 proxies IP addresses configured on redirected ports, the switch can redirect client requests to servers located on any subnet, anywhere.
- The switch can perform transparent substitution for all source and destination addresses, including destination port remapping. This provides support for comprehensive, fully-transparent proxies. These proxies are transparent to the user. No additional client configuration is needed.

The following procedure can be used for configuring proxy IP addresses:

1. Add proxy IP addresses to the redirection ports.

Each of the ports using redirection filters require proxy IP addresses to be configured. Each proxy IP address must be unique on your network. These are configured as follows:

>> SLB port 3# /cfg/slb/port 5	<i>(Select network port #5)</i>
>> SLB port 5# pip 200.200.200.68	<i>(Set proxy IP address for port #5)</i>
>> SLB port 5# proxy ena	<i>(Enable proxy port #5)</i>
>> SLB port 5# ../port 6	<i>(Select network port #6)</i>
>> SLB port 6# pip 200.200.200.69	<i>(Set proxy IP address for port #6)</i>
>> SLB port 6# proxy ena	<i>(Enable proxy port #6)</i>

2. If VMA is enabled, add proxy IP addresses for all other switch ports (except port 9).

Virtual Matrix Architecture (VMA) is normally enabled on the switch. In addition to enhanced resource management, this feature eliminates many of the restrictions found in earlier versions of the Web OS. It does require, however, that when any switch port is configured with an IP proxy address, all ports must be configured with IP proxy addresses. Otherwise, if VMA is disabled, only the client port with filters need proxy IP addresses and this step can be skipped.

The following commands can be used to configure the additional unique proxy IP addresses:

```
>> SLB port 6# ../port 1 (Select network port #1)
>> SLB port 1# pip 200.200.200.70 (Set proxy IP address for port #1)
>> SLB port 1# ../port 2 (Select network port #2)
>> SLB port 2# pip 200.200.200.71 (Set proxy IP address for port #2)
>> SLB port 2# ../port 3 (Select network port #3)
>> SLB port 3# pip 200.200.200.72 (Set proxy IP address for port #3)
>> SLB port 3# ../port 4 (Select network port #4)
>> SLB port 4# pip 200.200.200.73 (Set proxy IP address for port #4)
>> SLB port 4# ../port 7 (Select network port #7)
>> SLB port 7# pip 200.200.200.74 (Set proxy IP address for port #7)
>> SLB port 7# ../port 8 (Select network port #8)
>> SLB port 8# pip 200.200.200.75 (Set proxy IP address for port #8)
```

NOTE – Port 9 does not require a proxy IP address with VMA enabled.

See the *Web OS 8.3 Command Reference* for more information (/cfg/slb/adv/matrix).

3. 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 *Web OS 8.3 Command Reference*.

4. Apply and save your changes.

5. Check server statistics to verify that traffic has been redirected based on filtering criteria:

```
>> # /info/slb/group <group#>/filter <filter#> (View statistics for server filter)
```

Excluding Non-Cacheable Sites

Some Web sites provide content which isn't well-suited for redirection to cache servers. Such sites might provide browser-based games, 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 which allows this specific traffic to pass normally through the switch. This filter must have a higher precedence (a lower filter number) than the Application Redirection filter.

For example, if you wished 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>

Defining IP Address Ranges for Filters

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 `sip` (source) or `dip` (destination) defines the base IP address in the desired range, and the `smask` (source) or `dmask` (destination) is the mask which 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 `dip`.

As another example, you could configure the switch 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 3-2 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

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 SLB Options” on page 33](#) for details.

CHAPTER 4

Firewall Load Balancing

Network security has become an increasing concern for Web hosts and Internet Service Providers (ISPs). As a result, firewalls are commonly used to prevent unauthorized access to network resources. While firewalls are very effective at preventing network intrusions, they are often single points of failure and can reduce network availability.

Firewall Load Balancing (FWLB) with WebSwitches from Alteon WebSystems divides the load between many discrete firewalls, allowing 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.

Overview

Typically, a firewall server is inserted into the data path between the private (protected) network and the public network.

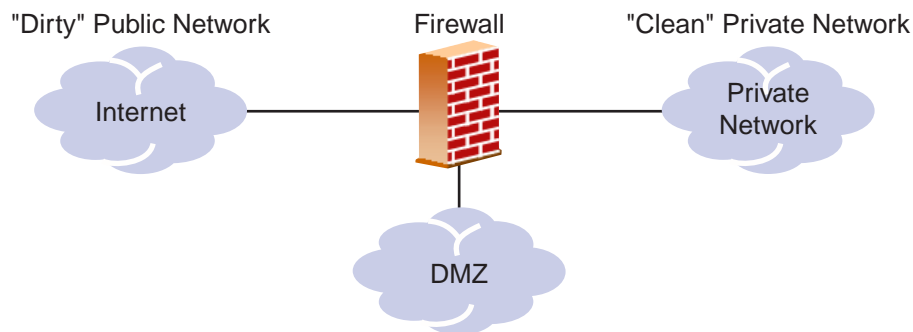


Figure 4-1 Typical Firewall Configuration Before FWLB

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.

All network traffic passing between the dirty and clean networks must traverse the firewall, which examines each individual packet. With firewall load balancing, filters that redirect all incoming IP traffic are configured on WebSwitches on both the dirty and clean sides of the firewall. On each switch, the filtered traffic is redirected in such a way as to load balance it across the firewalls. Because the switches intelligently maintain state information about the traffic flowing through them, they ensure that all traffic between specific IP source/destination address pairs flows through the same firewall. This, in turn, ensures that sessions established by the firewalls are maintained for their duration.

NOTE – While you can implement single-switch firewall load balancing, the switch will perform health checking on the firewall interfaces only.

Basic FWLB Implementation

The following figure shows a basic FWLB topology. This configuration requires a minimum of two WebSwitches; for extra redundancy, four switches and hubs can be used.

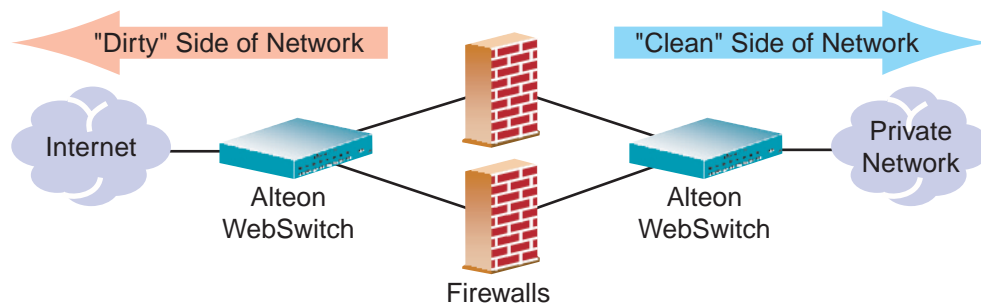


Figure 4-2 Basic FWLB Topology

The two firewalls being load balanced are in the middle of the network, separating the dirty side from the clean side. Two WebSwitches are also in place, one on the clean side of the firewalls and one on the dirty side.

NOTE – FWLB generally requires VLANs. If you will be using hubs and do not configure VLANs, you must enable Spanning Tree.

In the following two-switch configuration example, external clients connect to services at an IP address assigned to a virtual server on the clean-side WebSwitch. With Alteon WebSystems FWLB, data inbound or outbound to/from the Internet is load balanced among the available firewalls. Here's how the process works:

1. **The client request arrives at this network via the ingress port on the dirty-side WebSwitch.**
2. **A redirection filter balances incoming requests between two IP addresses.**

A filter is configured on the ingress port which will redirect requests to a real server group that consists of two different IP addresses. Each IP address represents an IP interface on a different subnet on the clean-side switch on the other side of the firewall.

3. **Requests are routed to the firewalls.**

Two static routes are configured on the dirty-side switch. The first static route leads to one of the clean-side switch IP interfaces using one firewall as the next hop. The second static route leads to the other clean-side IP interface using the other firewall as the next hop. By combining the redirection filter and static routes, the client requests are load balanced between both firewalls.

Because WebSwitches intelligently maintain state information about the traffic flowing through them, they ensure that 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.

4. **The firewall decides if it should allow the packet and, if so, where to forward it.**

Client requests are forwarded or discarded according to rule sets configured for each firewall.

NOTE – Rule sets must be consistent across all firewalls.

Forwarded packets are sent on to the original destination address: the virtual server on the clean-side switch, where they are load balanced to the real servers using standard SLB configuration.

5. **The server response arrives at the ingress port on the clean-side WebSwitch.**
6. **Redirection filters balance responses among two IP addresses.**

Redirection filters are needed on all ingress ports on the clean-side WebSwitch that attach to real servers or internal clients on the clean-side of the network. Filters on these ports redirect the outbound traffic to a real server group on the dirty-side switch that consists of two different IP addresses. Each IP address represents an IP interface on two more subnets on the dirty-side switch on the other side of the firewall.

WebSwitches intelligently maintain state information. If NAT is not being used, all traffic between specific IP source/destination address pairs flows through the same firewall.

7. Outbound traffic is routed to the firewalls.

Two static routes are configured on the clean-side switch. The first static route leads to one of the dirty-side IP interfaces, using one firewall as the next hop. The second static route leads to the other dirty-side IP interface, using the other firewall as the next hop.

8. The firewall decides if it should allow the packet and, if so, where to forward it.

Each firewall forwards or discards the client requests according to the rules that are configured for it. Forwarded packets are sent on to the dirty-side WebSwitch, and from there, out to their destination on the Internet.

Configuration for Basic FWLB - an Example

The steps for configuring basic firewall load balancing are provided below. While two or four switches can be used, the following procedure assumes a network topology having one WebSwitch on either side of the firewalls.

Configure the Dirty-Side WebSwitch

1. Define the dirty-side IP interfaces.

There should be one dirty-side IP interface on a different subnet for each firewall being load balanced.

>> # /cfg/ip/if 1	(Select IP interface 1)
>> IP Interface 1# addr 192.168.13.1	(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 192.168.16.1	(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)

2. Configure real servers using the IP addresses of the clean-side IP interfaces.

Later in this procedure, you'll configure one clean-side IP interface on a different subnet for each firewall being load balanced. Create two real servers on the dirty-side switch, using the IP address of each clean-side IP interface.

```
>> IP Interface 2# /cfg/slb/real 1           (Select real server 1)
>> Real server 1# rip 192.168.14.1         (Assign clean-side IF 3 address)
>> Real server 1# ena                       (Enable real server 1)
>> Real server 1# ../real 2                (Select real server 2)
>> Real server 2# rip 192.168.18.1         (Assign clean-side IF 4 address)
>> Real server 2# ena                       (Enable real server 1)
```

NOTE – Each of the four firewall interfaces (two on each WebSwitch) in this example must be configured for a different IP subnet.

3. Place the 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               (Select real server 1 to group 1)
>> Real server group 1# add 2               (Select real server 2 to group 1)
```

4. Set the health check type for the real server group to ICMP.

```
>> Real server group 1# health icmp         (Select ICMP as health check type)
```

5. 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)
```

6. Enable server load balancing on the switch.

```
>> Real server group 1# /cfg/slb/on
```

7. Create the filters to allow local subnet traffic on the dirty side of the firewalls to reach the firewall interfaces.

Two filters are needed to prevent local traffic from being redirected to firewalls.

NOTE – Local traffic allowed includes VRRP updates and traffic between specific devices on the dirty side of the network.

>> Layer 4# /cfg/slb/filt 110	(Select filter 110)
>> Filter 110# sip any	(From any source IP address)
>> Filter 110# dip 192.168.14.0	(To this destination IP address)
>> Filter 110# action allow	(Allow frames with this DIP address)
>> Filter 110# ena	(Enable filter)
>> Filter 110# ../filt 120	(Select filter 120)
>> Filter 120# sip any	(From any source IP address)
>> Filter 120# dip 192.168.18.0	(To this destination IP address)
>> Filter 120# action allow	(Allow the traffic)
>> Filter 120# ena	(Enable filter)

8. Create the 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 a clean-side switch.

>> Filter 120# ../filt 223	(Select filter 223)
>> Filter 223# sip any	(From any source IP address)
>> Filter 223# dip any	(To any destination IP address)
>> Filter 223# proto any	(For any protocol)
>> Filter 223# action redir	(Perform redirection)
>> Filter 223# group 1	(To real server group 1)
>> Filter 223# ena	(Enable the filter)

9. Add filters to the ingress port.

>> Filter 223# ../port 5	(Select the ingress port)
>> SLB Port 5# add 110	(Add the filter to the ingress port)
>> SLB Port 5# add 120	(Add the filter to the ingress port)
>> SLB Port 5# add 223	(Add the filter to the ingress port)
>> SLB Port 5# filt ena	(Enable filtering on the port)

10. Define static routes to the clean-side IP interfaces, using the firewalls as gateways.

One static route is required for each firewall being load balanced. In this case, two paths are required: one that leads to clean-side IP interface 3 (192.168.14.1) through the first firewall (10.1.1.10) as its gateway, and one that leads to clean-side IP interface 4 (192.168.18.1) through the second firewall (10.1.2.10) as its gateway.

```
>> SLB Port 5# /cfg/ip/route
>> IP Static Route# add 192.168.14.1 255.255.255.255 10.1.1.10
>> IP Static Route# add 192.168.18.1 255.255.255.255 10.1.2.10
```

11. Apply and save the configuration changes

```
>> # apply
>> # save
```

Configure the Clean-Side WebSwitch

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 5) prevents server-to-server traffic from being redirected.

>> # /cfg/ip/if 3	<i>(Select IP interface 3)</i>
>> IP Interface 3# addr 192.168.14.1	<i>(Set the IP address for interface 3)</i>
>> IP Interface 3# mask 255.255.255.0	<i>(Set subnet mask for interface 3)</i>
>> IP Interface 3# ena	<i>(Enable IP interface 3)</i>
>> IP Interface 3# ../if 4	<i>(Select IP interface 4)</i>
>> IP Interface 4# addr 192.168.18.1	<i>(Set the IP address for interface 4)</i>
>> IP Interface 4# mask 255.255.255.0	<i>(Set subnet mask for interface 4)</i>
>> IP Interface 4# ena	<i>(Enable IP interface 2)</i>
>> IP Interface 4# ../if 5	<i>(Select IP interface 5)</i>
>> IP Interface 5# addr 192.168.20.1	<i>(Set the IP address for interface 5)</i>
>> IP Interface 5# mask 255.255.255.0	<i>(Set subnet mask for interface 5)</i>
>> IP Interface 5# ena	<i>(Enable IP interface 5)</i>

2. Configure real servers that have IP addresses of the dirty-side IP interfaces.

You should already have configured a dirty-side IP interface on a different subnet for each fire-wall being load balanced. Create two real servers on the clean-side switch, using the IP address of each dirty-side IP interface.

```
>> IP Interface 5# /cfg/slb/real 1           (Select real server 1)
>> Real server 1 # rip 192.168.13.1        (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 192.168.16.1        (Assign dirty-side IF 2 address)
>> Real server 2 # ena                     (Enable real server 2)
```

NOTE – Each of the four IP interfaces (two on each WebSwitch) 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                (Select real server group 1)
>> Real server group 1# add 1                (Select real server 1 to group 1)
>> Real server group 1# add 2                (Select real server 2 to group 1)
```

4. Set the health check type for the real server group to ICMP.

```
>> Real server group 1# health icmp          (Select ICMP as health check type)
```

5. 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)
```

6. Enable server load balancing on the switch.

```
>> Real server group 1# /cfg/slb/on
```


7. Create a filter to prevent server-to-server traffic from being re-directed.

```
>> Layer 4# /cfg/slb/filt 100           (Select filter 100)
>> Filter 100# sip any                 (From any source IP address)
>> Filter 100# dip 192.168.20.0       (To base IP address for IF 5)
>> Filter 100# dmask 255.255.255.0   (For the range of addresses)
>> Filter 100# proto any              (For any protocol)
>> Filter 100# action allow           (Allow traffic)
>> Filter 100# ena                   (Enable the filter)
```

8. Create the filters to allow local subnet traffic.

Two filters are needed to prevent local traffic from being redirected to firewalls.

```
>> Layer 4# ../filt 110                (Select filter 110)
>> Filter 110# sip any                 (From any source IP address)
>> Filter 110# dip 192.168.13.0       (To this destination IP address)
>> Filter 110# action allow           (Allow frames with this DIP address)
>> Filter 110# ena                   (Enable filter)
>> Layer 4# ../filt 120                (Select filter 120)
>> Filter 120# sip any                 (From any source IP address)
>> Filter 120# dip 192.168.16.0       (To this destination IP address)
>> Filter 120# action allow           (Allow the traffic)
>> Filter 120# ena                   (Enable filter)
```

9. 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 120# ../filt 223             (Select filter 223)
>> Filter 223# sip any                 (From any source IP address)
>> Filter 223# dip any                 (To any destination IP address)
>> Filter 223# proto any              (For any protocol)
>> Filter 223# action redir           (Perform redirection)
>> Filter 223# group 1                (To real server group 1)
>> Filter 223# ena                   (Enable the filter)
```

10. Add the filters to the ingress ports for the outbound packets.

Redirection filters are needed on all the ingress ports on the clean-side WebSwitch. 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 WebSwitch on port 2 and port 6.

```
>> Filter 223# ../port 2                (Select ingress port 2)
>> SLB Port 2# add 100                  (Add the filter to the ingress port)
>> SLB Port 2# add 110                  (Add the filter to the ingress port)
>> SLB Port 2# add 120                  (Add the filter to the ingress port)
>> SLB Port 2# add 223                  (Add the filter to the ingress port)
>> SLB Port 2# filt ena                 (Enable filtering on the port)
>> SLB Port 2# ../port 6                (Select ingress port 6)
>> SLB Port 6# add 100                  (Add the filter to the ingress port)
>> SLB Port 6# add 110                  (Add the filter to the ingress port)
>> SLB Port 6# add 120                  (Add the filter to the ingress port)
>> SLB Port 6# add 223                  (Add the filter to the ingress port)
>> SLB Port 6# filt ena                 (Enable filtering on the port)
```

11. Define static routes to the dirty-side IP interfaces, using the firewalls as gateways.

One static route is required for each firewall being load balanced. In this case, two paths are required: one that leads to dirty-side IP interface 1 (192.168.13.1) through the first firewall (10.1.3.10) as its gateway, and one that leads to dirty-side IP interface 2 (192.168.16.1) through the second firewall (10.1.4.10) as its gateway.

```
>> SLB Port 5# /cfg/ip/route
>> IP Static Route# add 192.168.13.1 255.255.255.255 10.1.3.10
>> IP Static Route# add 192.168.16.1 255.255.255.255 10.1.4.10
```

NOTE – Configuring static routes for FWLB does not require IP forwarding to be turned on.

12. Apply and save the configuration changes

```
>> # apply
>> # save
```

Adding a DMZ

Using a De-Militarized Zone (DMZ) in conjunction with firewall load balancing, the WebSwitch does the traffic filtering, off-loading this task from the firewall. A DMZ is created by configuring firewall load balancing on the same switch with another real server group and a redirection filter towards the DMZ subnets. Implementing a DMZ in conjunction with firewall load balancing enables the WebSwitch to do the traffic filtering, off-loading this task from the firewall.

The DMZ servers can be connected to the WebSwitch on the dirty side of the firewall. A typical firewall load balancing configuration with a DMZ is shown in the following illustration.

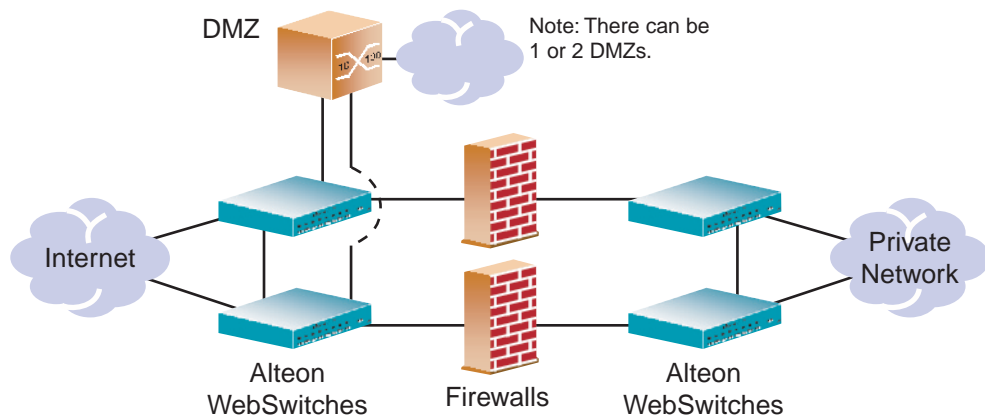


Figure 4-3 Typical Firewall Load-Balancing Topology with DMZ

The DMZ servers can be attached to the WebSwitch directly, or through an intermediate hub or switch. The WebSwitch 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 WebSwitch 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 as follows:

1. **On the dirty-side WebSwitch, create the filter to allow HTTP traffic to the DMZ Web servers.**

>> # /cfg/slb/filt 80	<i>(Select filter 80)</i>
>> Filter 80# sip any	<i>(From any source IP address)</i>
>> Filter 80# dip 205.178.29.0	<i>(To the DMZ base destination)</i>
>> Filter 80# dmask 255.255.255.0	<i>(For the range of DMZ addresses)</i>
>> Filter 80# proto tcp	<i>(For TCP protocol traffic)</i>
>> Filter 80# sport any	<i>(From any source port)</i>
>> Filter 80# dport http	<i>(To an HTTP destination port)</i>
>> Filter 80# action allow	<i>(Allow the traffic)</i>
>> Filter 80# ena	<i>(Enable the filter)</i>

2. **Create another filter to deny all other traffic to the DMZ Web servers.**

>> Filter 80# ../filt 89	<i>(Select filter 89)</i>
>> Filter 89# sip any	<i>(From any source IP address)</i>
>> Filter 89# dip 205.178.29.0	<i>(To the DMZ base destination)</i>
>> Filter 89# dmask 255.255.255.0	<i>(For the range of DMZ addresses)</i>
>> Filter 89# proto any	<i>(For TCP protocol traffic)</i>
>> Filter 89# action deny	<i>(Allow the traffic)</i>
>> Filter 89# ena	<i>(Enable the filter)</i>

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 5	<i>(Select the ingress port)</i>
>> SLB Port 5# add 80	<i>(Add the allow filter)</i>
>> SLB Port 5# add 89	<i>(Add the deny filter)</i>

4. **Apply and save the configuration changes.**

>> SLB Port 5# apply
>> SLB Port 5# save

Firewall Health Checks

Basic FWLB health checking, as discussed below, is automatic. No special configuration is necessary unless you wish to tune the health checking parameters. See [Chapter 10, “Health Checking”](#) for details.

Firewall Service Monitoring

To maintain high availability, WebSwitches monitor firewall health status and send packets only to healthy firewalls. There are two methods of firewall service monitoring: ICMP and HTTP. Each WebSwitch monitors the health of the firewalls on a regular basis by pinging the IP interfaces configured on its partner WebSwitch on the other side of the firewall.

If a WebSwitch 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 WebSwitch stops routing traffic to that interface, instead distributing it across the remaining, healthy WebSwitch interfaces and firewalls.

When a WebSwitch interface is in the “Server Failed” state, its partner WebSwitch continues to send pings to it at a user-configurable rate. After a specified number of successful pings, the interface (and its associated firewall) is brought back into service.

NOTE – To configure the switch to allow one-second intervals between health checks/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 x/inter 1/retry 2/restr 4`.

Physical Link Monitoring

WebSwitches 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 a WebSwitch 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, WebSwitches 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. 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 224 (Select filter 224)
>> Filter 224# proto tcp (For TCP protocol traffic)
>> Filter 224# action allow (Allow the traffic)
>> Filter 224# group 1 (Set real server group for redirection)
>> Filter 224# rport http (Set real server port for redirection)
>> Filter 224# ena (Enable the filter)
```

NOTE – Make sure that the number of each real filter is lower than the number of the “dummy” redirect filter.

Spanning Tree

Spanning Tree protocol ensures that there are no loops within a network topology. By avoiding the use of Spanning Tree, you can also avoid the lengthy delays that can occur (because of the protocol’s sequence of listening, learning, then forwarding or blocking) while STP is resolving itself. You can avoid Spanning Tree by using VLANs.

FWLB Checklist

NOTE – *Before* you configure any switches, make sure that the firewalls work as firewalls.

Primary Switch

- Create the switch interfaces. Add the data interface first. If needed, add to VLAN.
- Add the real servers. Make appropriate ones backup, if needed. Remember, backup real servers must be in order. For example, if the real server is 1, then it's backup would be real server 2.
- Add the real servers to a group.
- Set load balancing metric to HASH.
- Set the health option to ICMP (or HTTP).
- Turn on SLB.
- Create VRRP addresses.
- Turn on VRRP.
- Create the static routes to the opposing switch's interfaces.
- Create the filters for VRRP multicast, local traffic ALLOW and everything else REDIR.
- Add filters to ingress port.
- Turn IP Forwarding ON = VLAN configuration;
Turn FRWD OFF = Single VLAN approach.
- Turn off (or tune) Spanning Tree.

Secondary Switch

- Create the switch interfaces.
- Turn on SLB.
- Copy the static routes from the primary switch.
- Turn on VRRP.
- SYNC from primary (/oper/slb/s) IP of secondary, execute from primary.

Firewall Configuration Examples

- Four-Subnet FireWall Load Balancing Using Alteon WebSwitches
- Four Switches, Two Routing Firewalls, with No Interior Hubs on [page 110](#)
- Configuring NAT on Solaris Firewalls on [page 117](#)
- NAT for Single Devices on the Clean-Side Network on [page 118](#)

Four Subnet FWLB Using Alteon WebSwitches

This example shows how to set-up a four-switch firewall load-balancing sandwich, using Alteon WebSwitches for the clean and dirty side switches, and two firewalls. Figure [Figure 4-4](#) shows the desired topology.

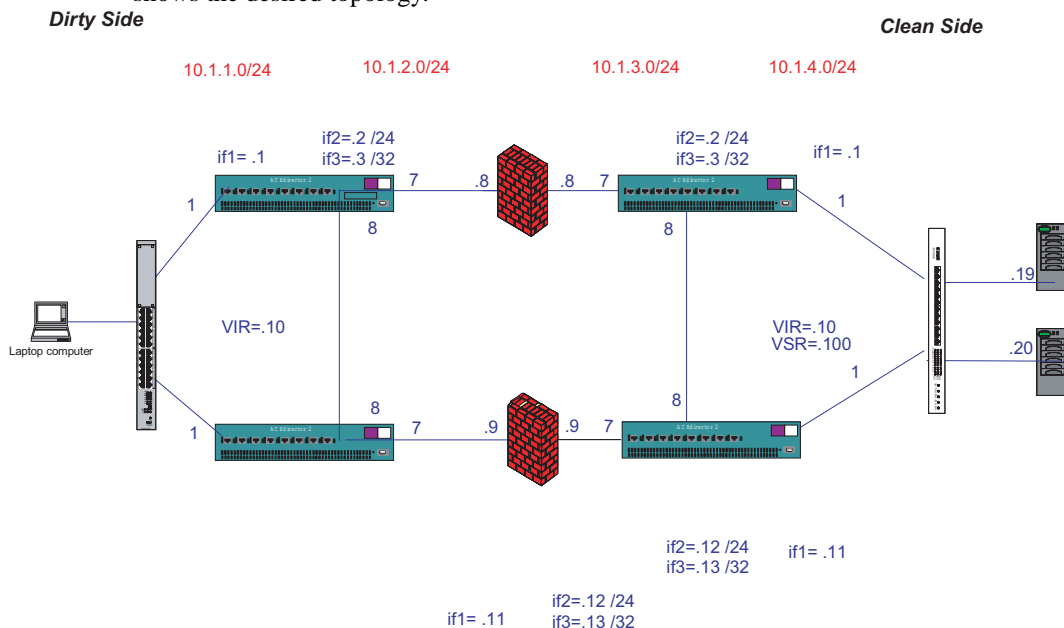


Figure 4-4 Four-Subnet FWLB with Alteon Switches

The procedure is divided into the following tasks:

- Set-up VLANs, IP addresses and static routes (with interface numbers) on all devices.
- Verify connectivity.
- On both the clean and dirty switches, create a group of real servers using the far switches' IP interfaces as RIPs, and set the metric for the group to hash.

- Create allow filters for VRRP and local traffic.
- Create a redirect filter for all traffic that needs to pass through the firewalls.
- Set-up SLB on the Alteon WebSwitches with two real servers in a group and assign the server groups to a virtual server.

NOTE – Before configuring firewall load balancing, ensure the two firewalls have been set-up for normal Layer 2/3 routing.

1. Configure Layer 2/3 routing on the primary dirty-side switch.

NOTE – When defining static routes, it is important to specify interface numbers.

```
>> # /cfg/vlan 2/add 7/add 8/ena

>> # /cfg/ip/if 1/mask 255.255.255.0/addr 10.1.1.1/ena
>> # /cfg/ip/if 2/vlan 2/mask 255.255.255.0/addr 10.1.2.2/ena
>> # /cfg/ip/if 3/vlan 2/mask 255.255.255.255/addr 10.1.2.3/ena

>> # /cfg/ip/frwd/on

>> # /cfg/ip/route
>> # add 10.1.3.2 255.255.255.255 10.1.2.8 2
>> # add 10.1.3.12 255.255.255.255 10.1.2.8 2
>> # add 10.1.3.3 255.255.255.255 10.1.2.9 3
>> # add 10.1.3.13 255.255.255.255 10.1.2.9 3

>> # /cfg/stp/off

>> # apply
>> # save
>> # /boot/reset
```

2. Configure Layer 2/3 routing on the secondary dirty-side switch.

```
>> # /cfg/vlan 2/add 7/add 8/ena

>> # /cfg/ip/if 1/mask 255.255.255.0/addr 10.1.1.11/ena
>> # /cfg/ip/if 2/vlan 2/mask 255.255.255.0/addr 10.1.2.12/ena
>> # /cfg/ip/if 3/vlan 2/mask 255.255.255.255/addr 10.1.2.13/ena
>> # /cfg/ip/frwd/on

>> # /cfg/ip/route
>> # add 10.1.3.2 255.255.255.255 10.1.2.8 2
>> # add 10.1.3.12 255.255.255.255 10.1.2.8 2
>> # add 10.1.3.3 255.255.255.255 10.1.2.9 3
>> # add 10.1.3.13 255.255.255.255 10.1.2.9 3

>> # /cfg/stp/off
>> #apply
>> #save
>> # /boot/reset
```

3. Configure Layer 2/3 routing on the primary clean-side switch.

```
>> # /cfg/vlan 2/add 7/add 8/ena

>> # /cfg/ip/if 1/mask 255.255.255.0/addr 10.1.4.1/ena
>> # /cfg/ip/if 2/vlan 2/mask 255.255.255.0/addr 10.1.3.2/ena
>> # /cfg/ip/if 3/vlan 2/mask 255.255.255.255/addr 10.1.3.3/ena

>> # /cfg/ip/frwd/on

>> # /cfg/ip/route
>> # add 10.1.2.2 255.255.255.255 10.1.3.8 2
>> # add 10.1.2.12 255.255.255.255 10.1.3.8 2
>> # add 10.1.2.3 255.255.255.255 10.1.3.9 3
>> # add 10.1.2.13 255.255.255.255 10.1.3.9 3

>> # /cfg/stp/off
>> # apply
>> # save
>> # /boot/reset
```

4. Configure Layer 2/3 routing on the secondary clean-side switch.

```

>> # /cfg/vlan 2/add 7/add 8/ena

>> # /cfg/ip/if 1/mask 255.255.255.0/addr 10.1.4.11/ena
>> # /cfg/ip/if 2/vlan 2/mask 255.255.255.0/addr 10.1.3.12/ena
>> # /cfg/ip/if 3/vlan 2/mask 255.255.255.255/addr 10.1.3.13/ena

>> # /cfg/ip/frwd/on

>> # /cfg/ip/route
>> # add 10.1.2.2 255.255.255.255 10.1.3.8 2
>> # add 10.1.2.12 255.255.255.255 10.1.3.8 2
>> # add 10.1.2.3 255.255.255.255 10.1.3.9 3
>> # add 10.1.2.13 255.255.255.255 10.1.3.9 3

>> # /cfg/stp/off
>> # apply
>> # save
>> # /boot/reset

```

5. Verify connectivity by pinging all interior interfaces.**6. Configure redirection on the primary dirty-side switch.**

```

>> # /cfg/slb/on
>> # /cfg/slb/real 1/rip 10.1.3.2/ena
>> # /cfg/slb/real 2/rip 10.1.3.3/ena
>> # /cfg/slb/real 3/rip 10.1.3.12/ena
>> # /cfg/slb/real 4/rip 10.1.3.13/ena

>> # /cfg/slb/group 1/add 1/add 2/add 3/add 4/metric hash

>> # /cfg/slb/filt 10/dip 10.1.1.0/dmask 255.255.255.0/ena
>> # /cfg/slb/filt 20/dip 224.0.0.0/dmask 255.255.255.0/ena
>> # /cfg/slb/filt 224/action redir/group 1/ena

>> # /cfg/slb/port 1/filt ena/add 10/add 20/ add 224

>> #apply
>> #save

```

7. Configure redirection on the primary clean-side switch.

NOTE – Because the sync operation causes a panic to occur, steps 6 and 7 will need to be repeated on the secondary switches.

```
>> # /cfg/slb/on
>> # /cfg/slb/real 1/rip 10.1.2.2/ena
>> # /cfg/slb/real 2/rip 10.1.2.3/ena
>> # /cfg/slb/real 3/rip 10.1.2.12/ena
>> # /cfg/slb/real 4/rip 10.1.2.13/ena

>> # /cfg/slb/group 1/add 1/add 2/add 3/add 4/metric hash

>> # /cfg/slb/filt 10/dip 10.1.4.0/dmask 255.255.255.0/ena
>> # /cfg/slb/filt 20/dip 224.0.0.0/dmask 255.255.255.0/ena
>> # /cfg/slb/filt 224/action redir/group 1/ena

>> # /cfg/slb/port 1/filt ena/add 10/add 20/ add 224
```

8. Set-up SLB on the clean-side switches (optional).

```
>> # /cfg/slb/real 5/rip 10.1.4.19/ena
>> # /cfg/slb/real 6/rip 10.1.4.20/ena
>> # /cfg/slb/group 2/add 5/add 6

>> # /cfg/slb/virt 1/ena/vip 10.1.4.100
>> # service 80/group 2

>> # /cfg/slb/port 1/server e
>> # /cfg/slb/port 7/client e
>> # /cfg/slb/port 8/client e
>> # apply
```

9. Configure VRRP on the clean-side switches.

```
>> # /cfg/vrrp/on
>> # /cfg/vrrp/vr 1/ena/addr 10.1.4.10/share dis/prio101
>> # track/ifs e

>> # /cfg/vrrp/vr 2/ena/vrid 2/addr 10.1.4.100/share dis/prio 101
>> # track/ifs e
>> # /cfg/vrrp/vr 3/ena/vrid 3/addr 10.1.3.10/if 2
>> # apply
>> # save
```

10. Configure VRRP on the dirty-side switches.**Primary Switch:**

```
>> # /cfg/vrrp/on
>> # /cfg/vrrp/vr 1/ena/prio101/addr 10.1.1.10/share dis
>> # track/l4pts e

>> # /cfg/vrrp/vr 2/ena/vrid 2/addr 10.1.2.10

>> # /cfg/slb/sync/prios d
>> # /cfg/slb/sync/peer 1/ena/addr 10.1.1.11
>> # apply
>> # save
>> # /oper/slb/sync
```

Secondary Switch:

```
>> # /cfg/vrrp/on
>> # /cfg/slb/sync/peer 1/ena/addr 10.1.1.1
```

11. Verify the FWLB “sandwich” is operational.

4 WebSwitches, 2 Routing Firewalls, with No Interior Hubs

In this configuration, all switches are active.

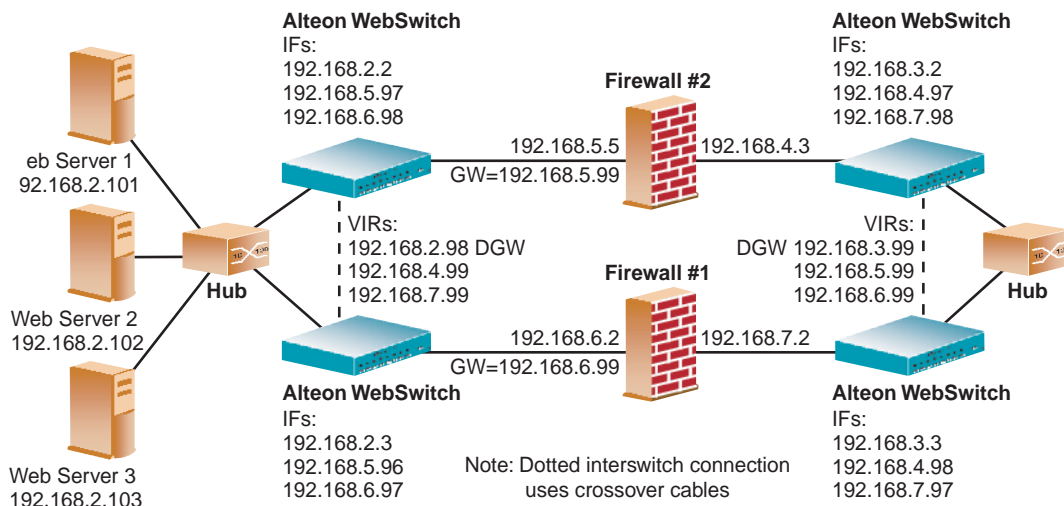


Figure 4-5 4 WebSwitches, 2 Routing Firewalls, with No Interior Hubs

NOTE – Throughout the following configuration dump, the sequence of commands is annotated in **bold**, to indicate what is being configured at each point in the command line interface.

The configuration is broken into the following tasks:

- Client-side primary switch configuration - [page 111](#)
- Client-side secondary switch configuration - [page 113](#)
- Server-side primary switch configuration - [page 114](#)
- Server-side secondary switch configuration - [page 116](#)

Client-Side Primary Switch Configuration

```

/* Port 5 is the ingress port for the client data.
/* (The use of port 5 is purely arbitrary. Any port can be used.)
/*
/cfg/port 5 /ena/tag d/pvid 2/pref gig/back fast
/cfg/port 5 /fast/speed any /fctl both/mode any /auto on
/cfg/port 5 /gig/fctl both/auto on
/*
/* VLAN 2 is a VLAN created to avoid Spanning Tree issues.
/*
/cfg/vlan 2/ena/name "VLAN 2"/jumbo disabled
/cfg/vlan 2/def 5
/cfg/vlan 1/ena/name "Default VLAN"/jumbo disabled
/cfg/vlan 1/def 1 2 3 4 6 7 8 9
/*
/* Interfaces.
/*
/cfg/ip/if 1/addr 192.168.3.2/mask 255.255.255.0/broad
192.168.3.255/ vlan 2/ena
/cfg/ip/if 2/addr 192.168.5.97/mask 255.255.255.0/broad
192.168.5.255/ vlan 1/ena
/cfg/ip/if 3/addr 192.168.6.98/mask 255.255.255.0/broad
192.168.6.255/ vlan 1/ena
/*
/* Routes to opposing-side switch for data flow.
/*
/cfg/ip/route/add 192.168.4.0 255.255.255.0 192.168.5.3
/cfg/ip/route/add 192.168.7.0 255.255.255.0 192.168.6.2
/*
/* Turn IP forwarding on.
/*
/cfg/ip/frwd/on
/*
/* VRRP addresses are used for health checking. This is for good
/* failover if a switch fails.
/*
/cfg/vrrp/on
/cfg/vrrp/track/ifs 10
/cfg/vrrp/vr 1/vrid 1/ena
/cfg/vrrp/vr 1/addr 192.168.3.99/ifs enabled
/cfg/vrrp/vr 3/vrid 3/if 2/ena
/cfg/vrrp/vr 3/addr 192.168.5.99/ifs enabled
/cfg/vrrp/vr 4/vrid 5/if 3/ena
/cfg/vrrp/vr 4/addr 192.168.6.99/ifs enabled
/*

```

```

/* Spanning Tree is turned off. If left on, it will "see" a loop.
/* Using a VLAN eliminates this problem.
/*
/cfg/stp/off
/*
/* Create real servers and real server groups and set-up FWLB health
/* checks
/* The inter/retry/restr values used below enable fast failover,
/* that is, one-second intervals between health checks/pings, two
/* failed pings to remove the firewall, and four successful pings
/* to restore the firewall to the real server group.
/cfg/slb/real 1/rip 192.168.4.99/inter 1/retry 2/restr 4/ ena
/cfg/slb/real 2/rip 192.168.7.99/inter 1/retry 2/restr 4/ena
/cfg/slb/group 1/metric hash/health icmp
/cfg/slb/group 1/add 1
/cfg/slb/group 1/add 2
/*
/* Turn SLB on.
/*
/cfg/slb/on
/*
/* Filter 50 allows multicast VRRP communications.
/*
/cfg/slb/filt 50/sip any/smask 0.0.0.0
/cfg/slb/filt 50/dip 224.0.0.0/dmask 255.255.255.0
/cfg/slb/filt 50/proto vrrp
/cfg/slb/filt 50/action allow/ena
/*
/* Allow local subnet.
/*
/cfg/slb/filt 100/sip any/smask 0.0.0.0
/cfg/slb/filt 100/dip 192.168.3.0/dmask 255.255.255.0
/cfg/slb/filt 100/proto any
/cfg/slb/filt 100/action allow/ena
/*
/* Redirect everything else through firewalls to opposing switch.
/*
/cfg/slb/filt 224/sip any/smask 0.0.0.0
/cfg/slb/filt 224/dip any/dmask 0.0.0.0
/cfg/slb/filt 224/proto any
/cfg/slb/filt 224/action redir/ena
/cfg/slb/filt 224/group 1/rport 0
/*

```



```

/*
/* Add the filters to port 5, the ingress port.
/*
/cfg/slb/port 5/filt enabled
/cfg/slb/port 5/add 50
/cfg/slb/port 5/add 60
/cfg/slb/port 5/add 70
/cfg/slb/port 5/add 100
/cfg/slb/port 5/add 224
/*

```

Client-Side Secondary Switch Configuration

The client-side secondary switch is configured with the same commands used for the primary client switch configuration. The only difference in configuration are the interfaces, as shown below:

```

/* Interfaces.
/*
/cfg/ip/if 1/addr 192.168.3.2/mask 255.255.255.0/broad 192.168.3.255/ vlan 2/ena
/cfg/ip/if 2/addr 192.168.5.97/mask 255.255.255.0/broad 192.168.5.255/ vlan 1/ena
/cfg/ip/if 3/addr 192.168.6.98/mask 255.255.255.0/broad 192.168.6.255/ vlan 1/ena
/*

```

Server-Side Primary Switch Configuration

```

/* Port 5 is the ingress port for the server data.
/* (The use of port 5 is purely arbitrary. Any port can be used.)
/*
/cfg/port 5 /ena/tag d/pvid 2/pref gig/back fast
/cfg/port 5 /fast/speed any /fctl both/mode any /auto on
/cfg/port 5 /gig/fctl both/auto on
/*
/* VLAN 2 is a VLAN created to avoid Spanning Tree issues.
/*
/cfg/vlan 2/ena/name "VLAN 2"/jumbo disabled
/cfg/vlan 2/def 5
/cfg/vlan 1/ena/name "Default VLAN"/jumbo disabled
/cfg/vlan 1/def 1 2 3 4 6 7 8 9
/*
/* Interfaces.
/*
/cfg/ip/if 1/addr 192.168.2.2/mask 255.255.255.0/broad
192.168.2.255/ vlan 2/ena
/cfg/ip/if 2/addr 192.168.4.97/mask 255.255.255.0/broad
192.168.4.255/ vlan 1/ena
/cfg/ip/if 3/addr 192.168.7.98/mask 255.255.255.0/broad
192.168.7.255/ vlan 1/ena
/*
/* Routes to opposing side switch for data flow.
/*
/cfg/ip/route/add 192.168.5.0 255.255.255.0 192.168.4.3
/cfg/ip/route/add 192.168.6.0 255.255.255.0 192.168.7.2
/*
/* Turn IP forwarding on.
/*
/cfg/ip/frwd/on
/cfg/vrrp/on
/cfg/vrrp/track/ifs 10
/cfg/vrrp/vr 1/vrid 1/ena
/cfg/vrrp/vr 1/addr 192.168.2.99/ifs enabled
/*
/* VRRP VSR for VIP failover
/*
/cfg/vrrp/vr 2/vrid 2/ena
/cfg/vrrp/vr 2/addr 192.168.2.200/ ifs enabled
/cfg/vrrp/vr 3/vrid 3/if 2/ena
/cfg/vrrp/vr 3/addr 192.168.4.99/ifs enabled
/cfg/vrrp/vr 4/vrid 5/if 3/ena
/cfg/vrrp/vr 4/addr 192.168.7.99/ifs enabled
/*

```

```

/*
/* Spanning Tree is turned off. If left on, it will "see" a loop.
/* Using a VLAN eliminates this problem.
/*
/cfg/stp/off
/*
/* Create real servers and real server groups and set-up FWLB health
/* checks
/cfg/slb/real 1/rip 192.168.5.99/inter 1/retry 2/restr 4/ ena
/cfg/slb/real 2/rip 192.168.6.99/inter 1/retry 2/restr 4/ena
/cfg/slb/real 3/rip 192.168.2.101/inter 1/retry 2/restr 4/ena
/cfg/slb/real 4/rip 192.168.2.102/inter 1/retry 2/restr 4/ena
/cfg/slb/real 5/rip 192.168.2.103/inter 1/retry 2/restr 4/ena
/cfg/slb/group 1/metric hash/health icmp
/cfg/slb/group 1/add 1
/cfg/slb/group 1/add 2
/cfg/slb/group 2/metric roundrobin/health http
/cfg/slb/group 2/add 3
/cfg/slb/group 2/add 4
/cfg/slb/group 2/add 5
/*
/* Virtual IP address for the Web servers.
/*
/cfg/slb/virt 1/vip 192.168.2.200 /ena
/cfg/slb/virt 1/service/http 2
/cfg/slb/virt 1/service/frag httpslb ena
/cfg/slb/port 1/client ena
/cfg/slb/port 5/server ena
/*
/* Turn SLB on.
/*
/cfg/slb/on
/*
/* Filter 50 allows multicast VRRP communications.
/*
/cfg/slb/filt 50/sip any/smask 0.0.0.0
/cfg/slb/filt 50/dip 224.0.0.0/dmask 255.255.255.0
/cfg/slb/filt 50/proto vrrp
/cfg/slb/filt 50/action allow/ena
/*
/* Allow local subnet.
/*
/cfg/slb/filt 100/sip any/smask 0.0.0.0
/cfg/slb/filt 100/dip 192.168.2.0/dmask 255.255.255.0
/cfg/slb/filt 100/proto any
/cfg/slb/filt 100/action allow/ena

```

```

/* Redirect everything else through the firewalls to the opposing
/* switch.
/*
/cfg/slb/filt 224/sip any/smask 0.0.0.0
/cfg/slb/filt 224/dip any/dmask 0.0.0.0
/cfg/slb/filt 224/proto any
/cfg/slb/filt 224/action redir/ena
/cfg/slb/filt 224/group 1/rport 0
/*
/* Add the filters for port 5, the ingress port.
/*
/cfg/slb/port 5/filt enabled
/cfg/slb/port 5/add 50
/cfg/slb/port 5/add 60
/cfg/slb/port 5/add 70
/cfg/slb/port 5/add 100
/cfg/slb/port 5/add 224
/*

```

Server-Side Secondary Switch Configuration

The server-side secondary switch is configured with the same commands used for the primary server switch configuration. The only difference in configuration are the interfaces, as shown below:

```

/* Interfaces.
/*
/cfg/ip/if 1/addr 192.168.2.2/mask 255.255.255.0/broad 192.168.2.255/ vlan 2/ena
/cfg/ip/if 2/addr 192.168.4.97/mask 255.255.255.0/broad 192.168.4.255/ vlan 1/ena
/cfg/ip/if 3/addr 192.168.7.98/mask 255.255.255.0/broad 192.168.7.255/ vlan 1/ena

/* All other parts are the same.

```

Firewall Configuration Example: 4 WebSwitches, 2 Routing Firewalls

Using this example, shown in [Figure 4-5 on page 110](#), the following firewall components are configured:

Firewall 1:	192.168.5.3 - Internet side; 192.168.4.3 - server side
Default Gateway:	192.168.5.99
Route Added:	192.168.2.0 via 192.168.4.99
Firewall 2:	192.168.6.2 - Internet side; 192.168.7.2 - server side
Default Gateway:	192.168.6.99
Route Added:	192.168.2.0 via 192.168.7.99

Configuring NAT on Solaris Firewalls

Network Address Translation (NAT) was set-up on the firewalls facing towards the dirty side of the set-up (205.178.17.0).

To configure NAT on Checkpoint firewalls, perform the following steps:

1. **Go to “Manage Objects.”**
2. **Select/Create the “Network” object.**
3. **From the NAT tab, select NAT.**
4. **Click on “Add rule automatically.”**
5. **Select Hide.**

This will hide a whole network behind a single IP address by using the range of TCP ports associated with the IP address.

6. **Enter the IP address you want to hide the network behind.**
7. **Publish the MAC address for this IP address.**

NOTE – Important! This enables the devices requesting the IP address on ingress back from the dirty side to get the IP address. The Solaris/NT command for publishing the MAC associated with a NAT’ed IP address is:

```
arp -s <NAT IP address> <MAC address of the dirty side NIC on firewall> pub
```

Example:

```
arp -s 205.178.17.100 00:50:fe:41:fe:c0 pub
```

8. **For NAT to work with FWLB, you must assign a different NAT IP address for each firewall.**

The default route for all firewalls was set to use 205.178.17.1. This works if you set an interface on your firewall that is on the 205.178.17.0 network. If not, it will point towards the VRRP address of the health check and the Alteon switch will then route to 205.178.17.0 network.

NAT for Single Devices on the Clean-Side Network

The following procedure describes how to NAT a single server or virtual server from the clean side of the firewall to the dirty side. To do this, you need to create another filter on the dirty side that does a redirect for the NAT'ed address towards the clean side. This filter should be placed before the local subnet allow and the final redirection filters.

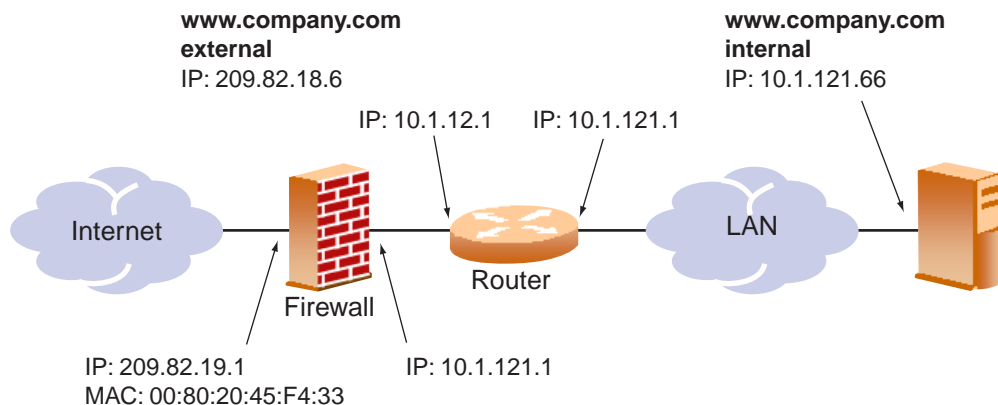


Figure 4-6 NAT for Single Devices on Clean-Side Network

You need to make the Web server at IP address 10.1.121.66 appear as 209.82.18.6 on the other side of the firewall. To do this, follow the steps below:

NOTE – All routing to the external and internal subnets should already be in place.

1. **Get the IP address of the system that you want to NAT (internal IP address).**

From the diagram in [Figure 4-6](#), this IP address is 10.1.121.66.

2. **Get the IP address of the system as it should be seen (external IP address).**

From the diagram in [Figure 4-6](#), this IP address is 209.82.18.6

3. **Add a routing statement.**

You need to tell the firewall which way to send packets meant for the Web server. Even though the firewall NATs all of the traffic, you still need to tell the system that the external IP address is to be routed through the internal router. Do this by adding the following line to the system boot scripts:

NOTE – If your firewall is multi-netted (two or more IP addresses per NIC), do not add the specific routing statement.

```
route add host 209.82.18.6 10.1.2.1 1
```

This tells the system that IP address 209.82.18.6 is to be routed though the router located at IP address 10.1.2.1. To have this command take effect right away, you can also type it directly on the command line.

Now, if you enter the following,

```
netstat -nr | grep 209.82.18.6
```

you should see the following line, indicating that the route has been successfully added.

```
209.82.18.6 10.1.2.1 UGH 0 0
```

4. Add Firewall-1 objects and NAT rules.

Now that the system is ready, you need to tell Firewall-1 what to do. Begin by creating two new “workstation” objects.

The first object is the “external” IP address (“www.company.com”), that is, how the company is known to the outside world. Enter the information as shown below:

Workstation Properties

SNMP | Encryption | Address Translation
General | Interfaces | Authentication

Name:

IP Address:

Comment:

Location: ☐ Internal ☒ External

Color:

Type: ☒ Host ☐ Gateway

☐ FireWall-1 installed
☐ Exportable

The next object is the “internal” IP address (“www.company.com.int”), that is, how the inside network knows the company. Enter the information as shown below:

The image shows the 'Workstation Properties' dialog box with the 'General' tab selected. The 'Name' field contains 'www.company.com.int'. The 'IP Address' field contains '10.1.121.66'. The 'Comment' field is empty. The 'Location' section has 'Internal' selected with a radio button. The 'Type' section has 'Host' selected with a radio button. There are checkboxes for 'FireWall-1 installed' and 'Exportable', both of which are unchecked. At the bottom are 'OK', 'Cancel', and 'Help' buttons.

Now that two objects are created, we need to set-up Firewall-1 to do the actual NAT. Go to the Address Translation Tab and enter the following:

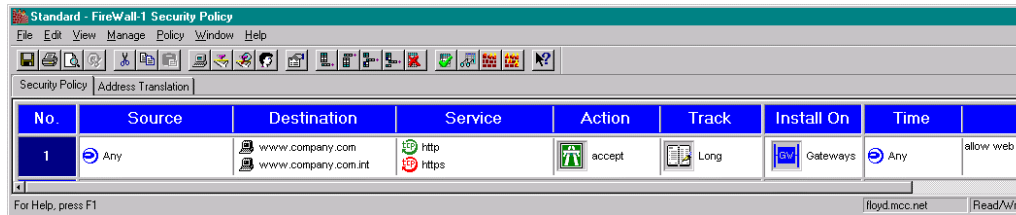
The image shows the 'Standard - FireWall-1 Security Policy' window with the 'Address Translation' tab selected. It displays a table with two rows of NAT rules. Row 1 shows a translation from 'Any' to 'www.company.com.int'. Row 2 shows a translation from 'www.company.com.int' to 'Any'. Both rules are marked as 'Original' and have 'Gateways' installed.

No.	Original Packet			Translated Packet			Install On
	Source	Destination	Service	Source	Destination	Service	
1	Any	www.company.com	Any	Original	www.company.com.int	Original	Gateways
2	Any	www.company.com.int	Any	Original	www.company.com	Original	Gateways

This sets up FW-1 to do the actual translation.

NOTE – In the actual rulebase where you want to deal with these new objects, you will probably have to enter **both** objects each time you want to put either of them into a rule. Without doing this, the traffic gets rejected.

The rules that use these new objects should look like this:



5. **Save and recompile the firewall rules.**
6. **Create a Virtual Interface Router (VIR) for the NAT'ed address.**
7. **Create a NAT rule on the other (clean-side) switch.**

This replaces the need to publish an ARP on each firewall (static NAT only).

Create a source NAT filter on the clean-side switch. This filter will change the IP address of the data coming through the firewalls and ensure that any data coming back from the statically NAT'ed device will return to the correct firewall, be un-Nat'ed, and sent back to the dirty-side requester.

```
>> # /cfg/slb/port 8/client ena/server dis/pip 192.168.2.10
>> # /cfg/slb/filt 85/sip any/smask 255.255.255.255
>> # /cfg/slb/filt 85/dip 10.0.1.0/dmask 255.255.255.255
>> # /cfg/slb/filt 85/proto tcp/sport any/dport any
>> # /cfg/slb/filt 85/log disabled/action nat/ena
>> # /cfg/slb/filt 85/nat source
```

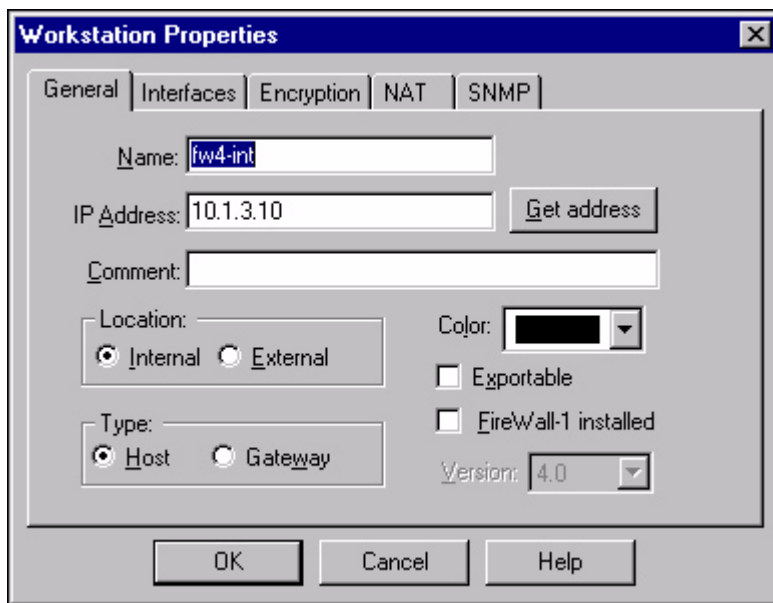
NOTE – You could make this a NAT rule on your CheckPoint firewall.

Static NAT, Performed by Firewall

This section describes how to perform source NAT and destination NAT at the same time to data coming in from the Internet (dirty side of network) to a virtual server/virtual server router on the inside (clean) side of the network.

1. **Get the IP address of the system that you want to NAT (internal IP address).**
2. **Get the IP address of the system as it should be seen (external IP address.)**
3. **Add Firewall-1 objects and NAT rules.**

Create workstation objects for the virtual server/virtual server router (TEP) and the interior interface of the firewall (fw4-int).



Now that we have created the two objects, we need to set-up a rule that creates a range of addresses which are outside of the clean-side network: in this example, 0.0.0.1 - >255.255.255.254 (Internet) is used. Go to the Address Translation Tab and enter the following.



No.	Original Packet			Translated Packet		
	Source	Destination	Service	Source	Destination	Service
1	internet	tep	Any	fw4-int	web	Original

4. Add a routing statement.

You need to add a host route for the statically NAT’ed IP address. In this example, IP address = 205.178.17.111. Do this by added the following line to the system boot scripts:

NOTE – To have this command take effect right away, you can also type it directly on the command line.

```
route add host 205.178.17.111 10.1.3.1
```

where IP 10.1.3.1 is the switch IP address that faces the target Web server at IP 205.178.13.123.

The destination NAT changes anything that is destined for IP 205.178.17.111 to IP 205.178.13.123.

5. Save and recompile the firewall rules.



CHAPTER 5

Virtual Private Network Load Balancing

The VPN (Virtual Private Network) load balancing feature in Web OS 8.3 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.

Overview

Virtual Private Networks

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, 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.

Using the hash algorithm on the source and destination address may not be possible in many VPN/firewall configurations 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 WebSwitch running Web OS 8.3 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 5-1](#). An external client is accessing an internal server. No Network Address Translation is performed by the VPN devices.

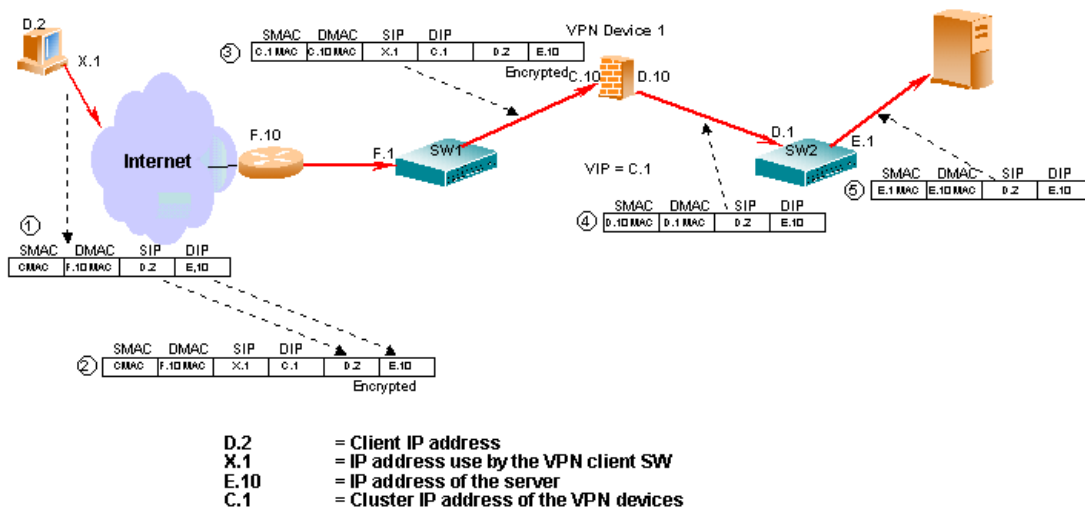


Figure 5-1 Basic Network Frame Flow and Operation

The basic steps that occur at the switches when a request arrives from the Internet are described below:

1. **The user prepares to send traffic to the destination server.**
2. **The VPN client SW encrypts the packet and sends it to the cluster IP address of the VPN devices.**
3. **Switch 1 (SW1) makes an entry in the session table and forwards the packet to VPN device 1.**

The selection of the VPN device is based on the hash load-balancing metric.

4. **VPN device strips the IP header and decrypts the encrypted IP header.**
5. **Switch 2 (SW2) forwards the packet to E.10.**

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 real VPN server, the switch adds an entry to the session table so that reverse traffic is redirected to the same VPN server. Finally, the frame is forwarded normally.

VPN Load-Balancing Configuration

Requirements

- Configure the switch with firewall load balancing. For more information, see “Firewall Load Balancing” in the Web OS Application Guide.
- Make sure VPN load balancing is enabled on the ports attached to the VPN devices, using the following command:

```
>> # /cfg/slb/port <port-number>/vpn ena
```

VPN Load Balancing Configuration Example

The following example uses Alteon WebSystems switches for VPN load balancing. The configuration is for four WebSwitches, four subnets, and two VPN devices.

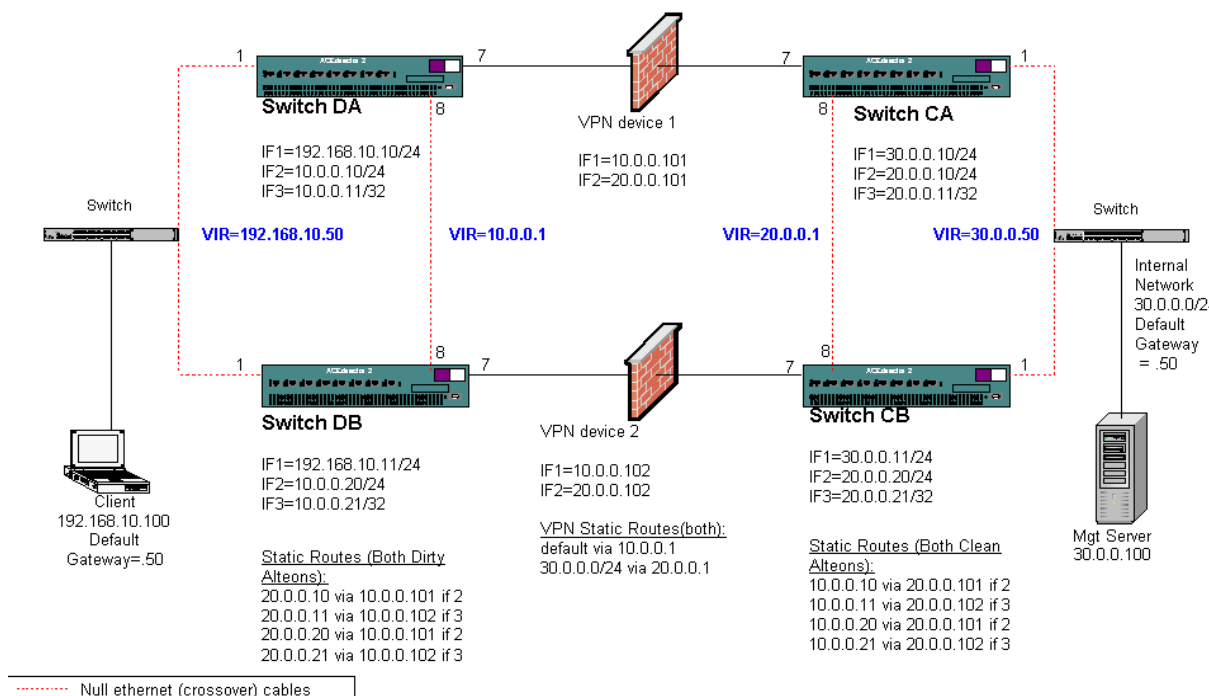


Figure 5-2 VPN Load-Balancing Configuration Example

Build the topology illustrated in [Figure 5-2 on page 128](#), and configure the switches as follows.

Configure the First Clean-Side Switch (CA)

1. Turn off bootp.

```
>> # /cfg/sys/bootp dis
```

2. Enable and define VLAN 2 for ports 7, 8, and 9.

```
>> # /cfg/vlan 2/ena/def 7 8 9
```

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           (Select IP interface 1 and enable)
>> IP Interface 1# mask 255.255.255.0   (Set subnet mask for interface 1)
>> IP Interface 1# addr 30.0.0.10      (Set IP address for interface 1)
>> IP Interface 1# vlan 1              (For VLAN 1)

>> IP Interface 1# ../if 2/ena         (Select IP interface 2 and enable)
>> IP Interface 2# mask 255.255.255.0   (Set subnet mask for interface 2)
>> IP Interface 2# addr 20.0.0.10      (Set IP address for interface 2)
>> IP Interface 2# vlan 2              (For VLAN 2)

>> IP Interface 2# ../if 3/ena         (Select IP interface 3 and enable)
>> IP Interface 3# mask 255.255.255.255 (Set subnet mask for interface 3)
>> IP Interface 3# addr 20.0.0.11      (Set IP address for interface 3)
>> IP Interface 3# vlan 2              (For VLAN 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
>> 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 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 VPN load balancing on the necessary ports.

```

>> # /cfg/slb/port 7/vpn ena          (Enable VPN LB on port 7)
>> # /cfg/slb/port 9/vpn ena          (Enable VPN LB on port 9)

```

11. Enable filter processing on the Server ports so that the responses from the Real Server will be looked up in the VPN session table.

```

>> # /cfg/slb/port 1/filter ena

```

12. Apply and save the configuration, and reboot the switch.

```

>> # apply
>> # save
>> # /boot/reset

```

Configure the Second Clean-Side Switch (CB)

1. Turn off bootp.

```
>> # /cfg/sys/bootp dis
```

2. Define and enable VLAN 2 ports 7 and 9.

```
>> # /cfg/vlan 2/ena/def 7 9
```

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/vl 2
>> # /cfg/ip/if 3/ena/mask 255.255.255.255/addr 20.0.0.21/vl 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 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 Server Load Balancing.

```

>> 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 VPN load balancing on the necessary ports.

```

>> Real server group 1# ../port 7/vpn ena
>> SLB port 7# ../port 9/vpn 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 9# ../port 1/filter ena
```

12. Apply and save the configuration, and reboot the switch.

```
>> SLB port 9# apply
>> SLB port 9# save
>> SLB port 9# /boot/reset
```

Configure the First Dirty-Side WebSwitch (DA)

1. Turn off bootp.

```
>> # /cfg/sys/bootp dis
```

2. Enable and define VLAN 2 for ports 7 and 8.

```
>> # /cfg/vlan 2/ena/def 7 8
```

3. Turn off Spanning Tree Protocol.

```
>> # /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 server load balancing.

```

>> 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 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
```

- 12. 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 224
>> # ena
>> # sip any
>> # dip any
>> # action redir
>> # ../filt 224/adv
>> # fwlb ena
```

- 13. Add filters to the ingress port.**

```
>> # ../port 1
>> # filt ena
>> # add 100/add 110/add 224
```

- 14. Apply and save the configuration, and reboot the switch.**

```
>> # apply
>> # save
>> # /boot/reset
```

Configure the Second Dirty-Side WebSwitch (DB)

1. Turn off bootp.

```
>> # /cfg/sys/bootp dis
```

2. Enable and define VLAN 2 for ports 7 and 8.

```
>> # /cfg/vlan 2/ena/def 7 8
```

3. Turn off spanning tree protocol.

```
>> # /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 server load balancing.

```

>> # /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
>> # /cfg/slb/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 224
>> # ena
>> # sip any
>> # dip any
>> # proto any
>> # action redir
>> # /cfg/slb/filt 224/adv
>> # fwlb ena
```

12. Add filters to the ingress port.

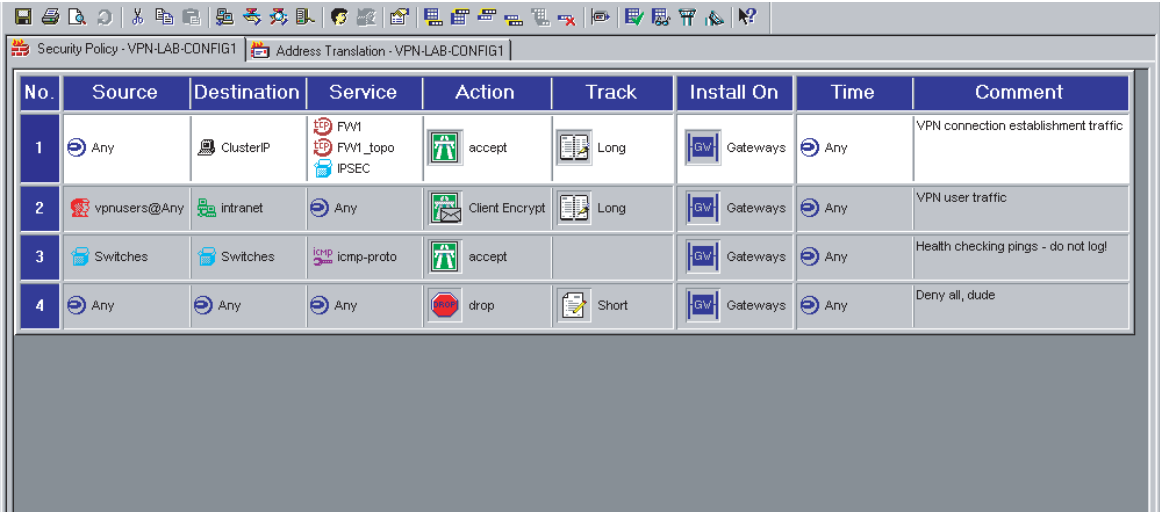
```
>> # /cfg/slb/port 1
>> # filt ena
>> # add 100/add 110/add 224
```

13. Apply and save the configuration and reboot the switch.

```
>> # apply
>> # save
>> # /boot/reset
```

Test Configurations and General Topology

The switches should be able to health check each other, and all switches should see four real servers up. (Rules on the VPN devices have to permit this – see [Figure 5-3 on page 141.](#))



The screenshot shows the Checkpoint Policy Editor interface. At the top, there are tabs for 'Security Policy - VPN-LAB-CONFIG1' and 'Address Translation - VPN-LAB-CONFIG1'. Below the tabs is a table with the following columns: No., Source, Destination, Service, Action, Track, Install On, Time, and Comment. The table contains four rules:

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 5-3 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)

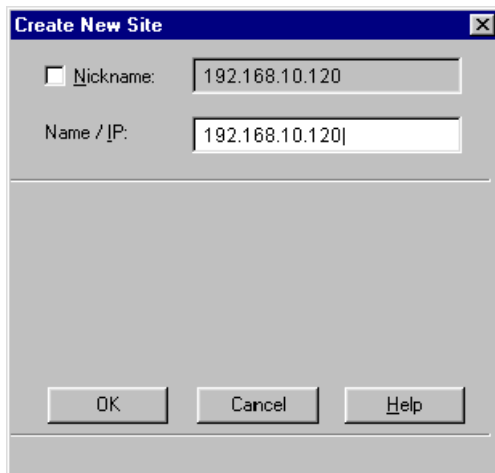
This should change the Virtual Router Redundancy Protocol (VRRP) preferences. You can view VRRP preferences using the CLI command `/info/vrrp`.

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

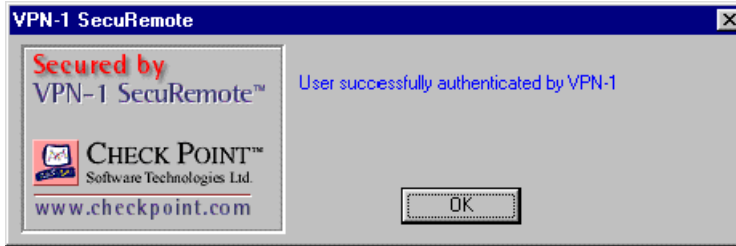
1. Launch the SecuRemote client on the dirty side of the network.
2. Add a new site.



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 asked to authenticate yourself.
6. Enter `vpnuser` for user name and `alteaon` for the password.



7. You will see a message verifying that you were authenticated.



8. Browse to the Website.

If there are other services running on other servers in the internal network, you should also be able to reach those services. All of this traffic is traveling over the VPN and is being 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 switches of the dirty side (that is, hashed on client IP address/Destination IP address), you can 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.

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 6

Global Server Load Balancing

This chapter provides a conceptual overview of Global Server Load Balancing (GSLB) and configuration examples for performing GSLB across multiple geographic sites. The number of supported GSLB sites per switch has been increased to 64, with a total aggregate of 2048 service/site combinations.

NOTE – Both the optional Server Load Balancing and Global Server Load Balancing software keys must be enabled. See the *Web OS 8.3 Command Reference* for details.

GSLB Overview

GSLB lets you balance server traffic load across multiple physical sites. This allows you to smoothly integrate the resources of a world-wide series of server sites and balance Web content (or other services) intelligently among them. Alteon WebSystems' GSLB system takes into account individual sites' health, response time, and geographic location for a global performance perspective.

NOTE – URL-based server load balancing is compatible with GSLB. Cookie-based persistence is compatible with GSLB using Active Cookie Mode (Cookie Rewrite Mode).

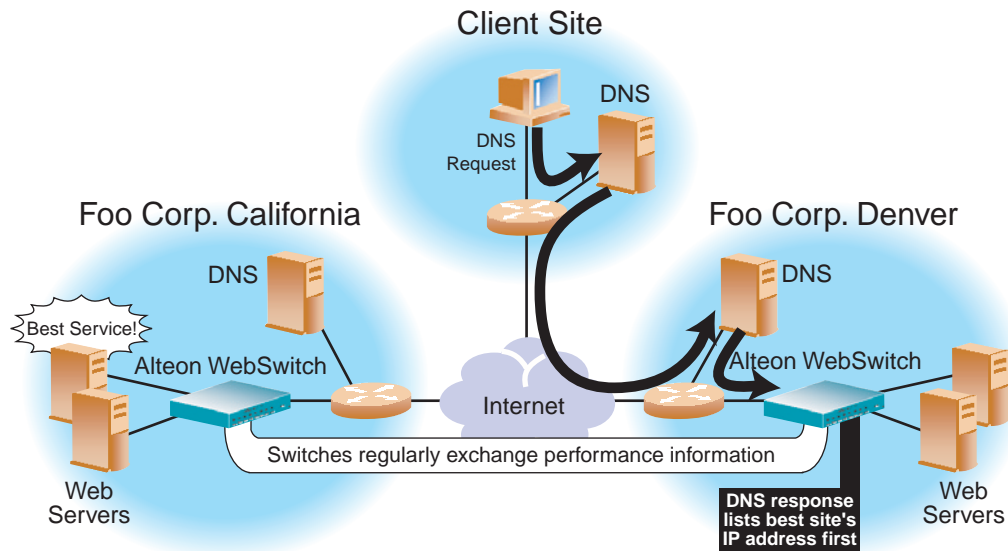
Benefits

GSLB meets the following demands for distributed network services:

- High content availability through distributed content and distributed decision making. If one site becomes disabled, the others become aware of it and take up the load.
- No latency during client connection set up. Instant site hand-off decisions can be made by any distributed switch.
- The best performing sites get a majority of traffic over a given period of time but are not overwhelmed.
- Switches at different sites regularly exchange information through 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.
- Takes geography as well as network topology into account.
- Gives creative control to the network administrator or webmaster to build and control content by user, location, target application, and more.
- Easy to deploy, manage, and scale. Switch configuration is straightforward. There are no complex system topologies involving routers, protocols, and so on.
- Provides flexible design options.
- Supports all IP protocols.

How GSLB Works

Consider the following sample network:



1. Browser requests `www.foo-corp.com` IP address from local DNS.
2. Client's DNS asks its upstream DNS, which in turn asks the next, and so on, until the address is resolved.
3. The Foo Corp. Denver DNS knows that the local Alteon WebSwitch is an authoritative name server for `www.foo-corp.com`.
4. The switch DSLB software knows that Foo Corp. California currently provides better service, and responds with Foo Corp. California's virtual IP address listed first.
5. The client connects to Foo Corp. California for the best service.

Figure 6-1 DNS Resolution with Global Server Load Balancing

A client is using their Web browser to view the website for the Foo Corporation at “`www.foo-corp.com`.” The Foo Corporation has two sites: one in California, and one in Denver, each with identical content and services available. Both sites have an Alteon WebSwitch configured for GSLB. These switches are also configured as the Authoritative Name Servers for “`www.foo-corp.com`.”

When a client loads their Web-browsing software and enters the URL for a website such as “www.foocorp.com,” a query is sent to the client’s local DNS server asking for the IP address representing the domain name entered. If the local DNS server does not have this information cached, it will in turn ask a DNS server further upstream. Eventually, the request will either reach an upstream DNS server that has this information on hand or reach one of the Foo Corporation’s DNS servers. The Foo Corporation’s DNS server has been configured to know that the local Alteon WebSystems switch with GSLB software is the authoritative name server for “www.foocorp.com.”

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-cruising needs. It can respond with a list of IP addresses for the Foo Corporation’s distributed sites, prioritized by performance, geography, and other criteria.

The client’s Web browser will use the IP address information 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 will issue an HTTP Redirect and transparently cause 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.

GSLB Configuration Example

Summary

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.
- 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 Domain Name System (DNS) server for each service 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 as usual 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, make each switch recognize its remote peers.
 - On each switch, configure a remote real server entry for each remote service.
 - Add the remote real server entry to an appropriate real server group.
 - Enable GSLB.

Example GSLB Configuration Procedure

Consider the following example network:

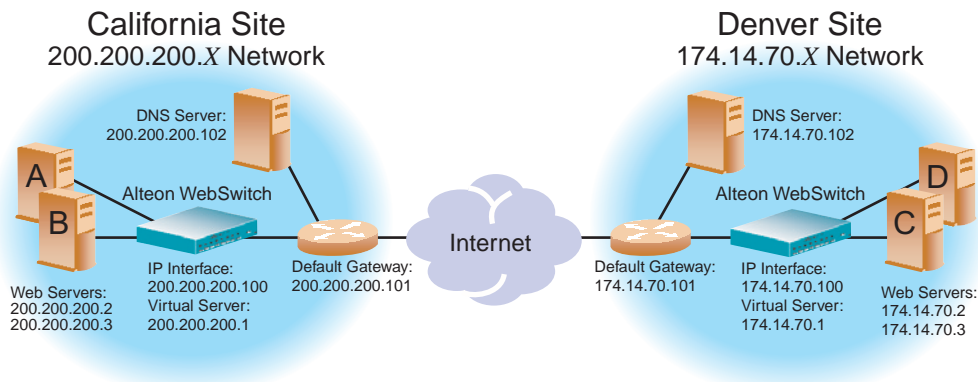


Figure 6-2 GSLB Example Topology

In the following examples, many of the options are left to their default values. See [“Additional SLB Options” on page 33](#) for more options.

The following is required prior to configuration:

- You must be connected to the switch command line interface 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 *Web OS 8.3 Command Reference*.

Part One: Configure the California Site with Basic System Items

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 Web OS Browser-Based Interface (BBI) also uses port 80. Both services cannot use the same port. If the Web-based interface is enabled (see the `/cfg/sys/http` command in Chapter 7 of the *Web OS 8.3 Command Reference*), configure it to use a different port.

For example, to change the BBI port to 8080, enter the following command:

>> # <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 is the entity that 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 the level of TCP/IP reachability of the real servers.

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

>> System# <code>/cfg/ip/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.**

In this example, a 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 hostname 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.foo corp.com.”

Part Two: 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 Considerations” on page 24](#).

For this example, the host real servers have IP addresses on the same IP subnet:

Table 6-1 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 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)
```

3. On the California switch, define a real server group.

The following commands 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 IP on a virtual server defined on the switch. Clients acquire the virtual IP 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 IP and is associated with our 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 Web service. Other TCP/IP services can be configured in a similar fashion. For a list of other well-known services and ports, see the command option information in the “Configuration” chapter of the *Web OS Command Reference*.

5. On the California switch, define the type of L4 traffic processing each port must support.

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

Table 6-2 GSLB Example: California Alteon 180 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)
```

Part Three: Configure the California Site for GSLB

1. On the California switch, define each remote site.

Add and enable the IP address for the IP interface of up to 64 remote sites. 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.

2. On the California switch, assign each remote distributed service to a local virtual server.

NOTE – This step can result in improper configuration if not clearly understood. Please take care to note where each configured value originates.

In this step, the local California site is configured 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's 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 will be configured with the IP address of the remote virtual server, rather than the usual IP address of a local physical server.

Also, the “remote” property will be enabled, and the real server entry will be added to the real server group under the local virtual server for the intended service. Finally, since the real server health checks will be 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	<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 – The IP address of the real server being added is taken from the virtual server IP address on the remote switch. Do not confuse this value with the IP interface address on the remote switch.

3. On the California switch, define the domain name and hostname for each service hosted on each virtual server.

In this example, the domain name for the Foo Corporation is “foocorp.com,” and the hostname 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 hostname)
```

If other services were defined (such as FTP), additional hostname entries would be made.

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)
```

Part Four: Configure the Denver Site with Basic System Items

Following the same procedures as above, configure the Denver site as follows.

1. If the Web OS BBI is to be used for managing the Denver switch, change its service port.

>> # /cfg/sys	<i>(Select the System Menu)</i>
>> System# wport 8080	<i>(Set service port 8080 for WBI)</i>

2. On the Denver switch, define an IP interface.

>> # /cfg/ip/if 1	<i>(Select IP interface #1)</i>
>> IP Interface 1# addr 174.14.70.100	<i>(Assign IP address for the interface)</i>
>> IP Interface 1# ena	<i>(Enable IP interface #1)</i>

3. On the Denver switch, define the default gateway.

>> IP Interface 1# ../gw 1	<i>(Select default gateway #1)</i>
>> Default gateway 1# addr 174.14.70.101	<i>(Assign IP address for the gateway)</i>
>> Default gateway 1# ena	<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.

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.foocorp.com.”

Part Five: 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 6-3 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 Server C IP address)
>> 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 Server D IP address)
>> 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 traffic processing that each port must support.

In this example, the following ports are being used on the Alteon 180 WebSwitch:

Table 6-4 Web Host Example: Alteon 180 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)

Part Six: Configure the Denver Site for GSLB

Following the same steps in Part Three, configure the Denver site as described below:

1. On the Denver switch, define each remote site.

Since we are now configuring the Denver site, Denver is local and California is remote. Add and enable the IP address for the IP interface of up to eight remote sites. 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:

```
>> Server Load Balancing# gslb/site 1           (Select Remote Site #1)
>> Remote site 1# prima 200.200.200.100         (Define remote IP interface address)
>> Remote site 1# ena                           (Enable remote site #1)
```

2. On the Denver switch, assign each remote distributed service to a local virtual server.

NOTE – This step can result in improper configuration if not clearly understood. Please take care to note where each configured value originates.

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's 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 will be configured with the IP address of the remote virtual server, rather than the usual IP address of a local physical server.

Also, the “remote” property will be enabled, and the real server entry will be added to the real server group under the local virtual server for the intended service. Finally, since the real server health checks will be 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 – The IP address of the real server being added is taken from the virtual server IP address on the remote switch. Do not confuse this value with the IP interface address on the remote switch.

3. On the Denver switch, define the domain name and hostname 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 hostname 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/hname www   (Define HTTP hostname)
```

4. On the Denver switch, turn on Global Server Load Balancing.

```
>> Virtual server 1# /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)
```


IP Proxy Addresses for Non-HTTP Application Redirects

Alteon WebSystems switches with Web OS software installed can configure GSLB remote servers to have any user request sent to them using a load-balancing mechanism called *IP Proxy*.

NOTE – This feature should be used as a method of last resort for GSLB implementations - in topologies where the remote servers are usually virtual IP addresses in other Alteon WebSystems switches.

How IP Proxy Works

Example: The figure below shows two GSLB sites, each with one local virtual server (VIP 1) serviced by two real servers in real server group 1. The applications being load balanced are HTTP and POP3. The network administrator wants to have any request that cannot be serviced locally to be sent to the peer site. HTTP requests will be sent to the peer site using HTTP Redirect. Any other application request will be sent to the peer site using the IP Proxy feature.

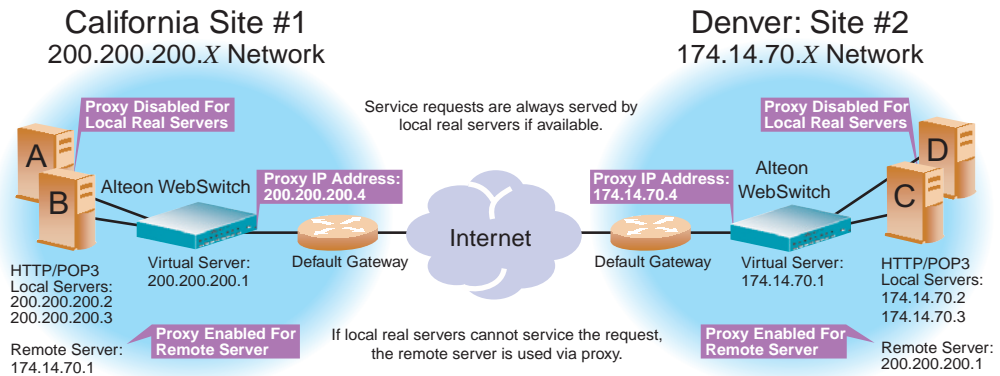


Figure 6-3 POP3 Request Fulfilled via IP Proxy

When the POP3 processes at site #1 terminate because of operator error, the following events occur to allow users' 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 switch rewrites the request, such that it now contains a proxy IP address as the IP source address (IPSA), and the virtual server IP address at site #2 as the IP destination address (IPDA).
3. The switch at Site #2 receives the TCP SYN (POP3) request to its virtual server that looks like a normal SYN frame, and thus, performs normal local load-balancing mechanisms.
4. The TCP SYN ACK coming from site #2's local real server IP address is sent back toward the IP address specified by the proxy IP address.
5. The switch at site #2 sends the TCP SYN ACK frame towards site #1, with site #2's virtual server IP address as the IP source address and site #1's proxy IP address as the IP destination address.
6. The switch at site #1 receives the frame and translates it, using site #1's virtual server IP address as the IP source address and the client's IP address as the IP destination address.

This cycle continues for the remaining frames that are necessary to transmit the client's mail, until a FIN frame is received.

Configuring IP Proxy

In keeping with the previous example, starting on [page 150](#), the switch at site #1 in California is configured with switch port 6 connecting to the default gateway and real server 3 representing the remote server in Denver. The following commands are used to configure the IP proxy on site #1 in California:

NOTE – If any port is configured with a proxy IP address, then all ports (except port 9) must be configured with a unique proxy IP address prior to enabling Virtual Matrix Architecture (VMA). Once they are configured, proxy IP addresses not in use can be disabled.

>> # /cfg/slb/port 6	(Select port to default gateway)
>> SLB port 6# pip 200.200.200.4	(Set unique proxy IP address)
>> SLB port 6# proxy enable	(Enable proxy for switch port 6)
>> SLB port 6 /cfg/slb/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)

If you want to configure IP Proxy on site #2, the following commands are issued on the Denver switch:

>> # /cfg/slb/port 5	(Select port to default gateway)
>> SLB port 5# pip 174.14.17.4	(Set unique proxy IP address)
>> SLB port 5# proxy enable	(Enable proxy for switch port 5)
>> SLB port 5# /cfg/slb/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)

Basic Tests for GSLB Operation

- Execute browser request to the configured service (“www.foo corp.com” in the example above).
- On each switch, examine the /info/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.
- On each switch, examine the following statistics:
 - /stats/slb/gslb/virt <virtual server number>
 - /stats/slb/gslb/group <real server group number>
 - /stats/slb/maint

GSLB Client Proximity Tables

In certain customer configurations, IANA data does not provide sufficient geographic separation of proximity information. As a result, large ISP partners cannot use their own geographic data to determine GSLB site selection based on client location. Web OS software supports client proximity tables using static “client to site” mapping. Switch managers can configure private client proximity information. The limit on the number of entries in the proximity database is 128.

The use of a static client/site database allows customizing for the user environment.

NOTE – The switch supports a single domain only.

Observe the following:

- No health checks or pings for virtual servers in the network table.
- Switch replies with only one virtual server IP address, based on response time and min-con value.

Configurable Source Network <--> Site Preference Tables,

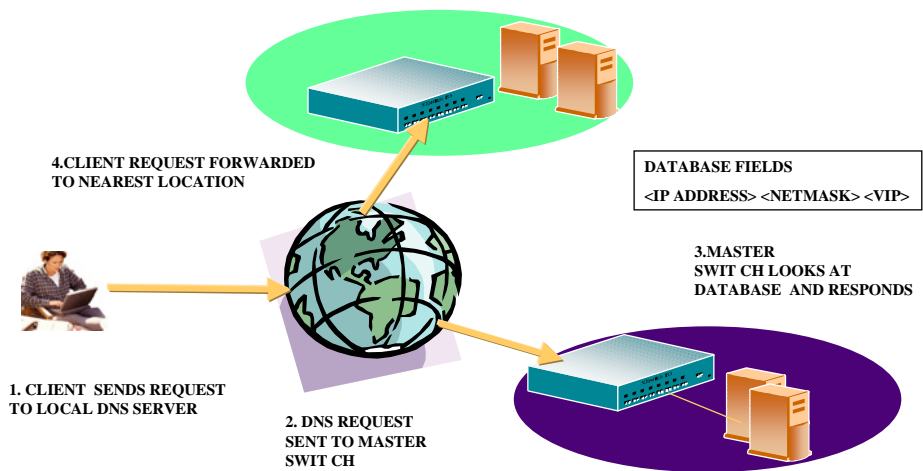


Figure 6-4 GSLB Proximity Tables: How They Work

GSLB Proximity Configuration Example

205.178.13.* prefers site 3, site 1

204.165.*.* prefers site 4, site 2

Here are the commands to configure this scenario:

```
>> # /cfg/slb/gslb/lookup/lookups ena
>> # dname alteon.com
>> # network 1
>> # sip 205.178.13.0
>> # mask 255.255.255.0
>> # vip1 IP addr of Site 3
>> # vip2 IP addr of Site 1
>> # ../network 2
>> # sip 204.165.0.0
>> # mask 255.255.255.0
>> # vip1 IP addr of Site 4
>> # vip2 IP addr of Site 2
```

Using this configuration, the DNS request “alteonwebsystems.com” from 205.178.13. will get a DNS response with only one virtual IP address; for example, site 1, if site 1 has less load than site 3.

CHAPTER 7

Content Intelligent Switching

The following table lists the primary topics described in this chapter and the page number where you'll find information about each feature.

Functionality	Features/Description	See
Content-Intelligent Server Load Balancing	URL Parsing/URL-Based WCR	page 170
	URL-Based Server Load Balancing	page 179
	Virtual Hosting	page 189
	Parsing based on browser type	page 192
	URL Hashing	page 196
	Preferential Treatment, based on Cookies	page 193
Content-Intelligent Web Cache Redirection	Cachability based on domain name	page 185
	Redirection based on domain name	page 185

NOTE – Virtual Matrix Architecture (VMA) should be enabled when using any content-intelligent switching feature. Prior to enabling VMA, you must either enable Direct Access Mode (DAM) or configure a (PIP) proxy IP address on all switch ports (except port 9).

Content Switching Overview

Working with session content is much more demanding than examining TCP/IP protocol 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.
- Parsing content requests temporarily terminates the TCP connection from a client. In other words, the WebSwitch must first pretend that it is the server, ask the client what it wants, examine the request, and then open a connection to an appropriate server. While this is happening, the WebSwitch must temporarily buffer the request, which consumes system memory. This temporary termination is called a *delayed binding*.
- With delayed binding, two independent TCP connections span a Web session: one from the client to the WebSwitch and the second from the WebSwitch to the selected server. The WebSwitch 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 WebSwitch, particularly when the switch must process thousands of these sessions simultaneously.

In addition to real-time traffic and connection processing, a content switch needs to monitor the 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 continue to process network protocols while failing to retrieve any content. Furthermore, if content is segregated in different servers or server farms, the WebSwitch 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.

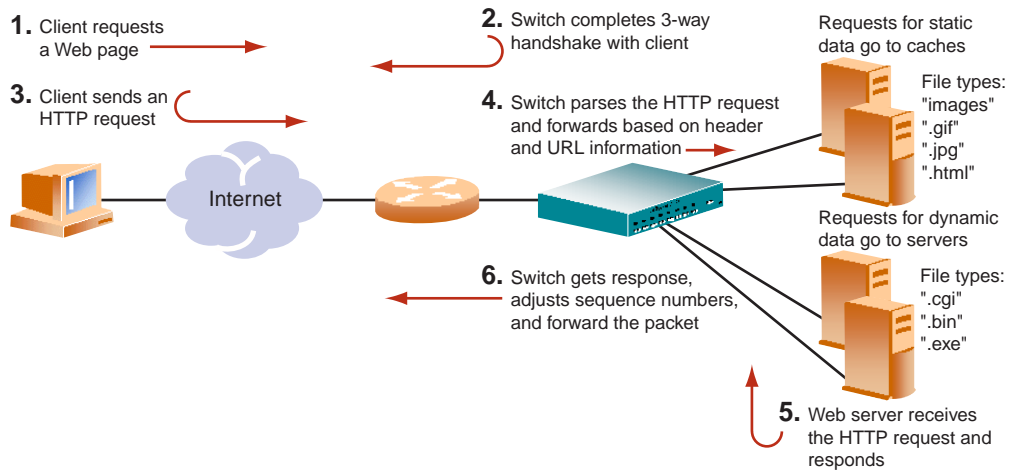


Figure 7-1 Content-Aware Load Balancing Example

To fulfill the requirements described above, a WebSwitch needs to perform numerous processing 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. These functions are executed whenever a new request arrives. In addition, the switch must perform background functions, such as updating network topology, health-checking servers, applications and server sites, measuring server performance, and so on, on a periodic basis.

URL-Based Web Cache Redirection

By separating static and dynamic content requests via URL parsing, Web OS 8.3 enables you to send requests with specific URLs or URL substrings to designated cache servers. The URL-based redirection option allows you to perform cache server farm tuning and offload 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 does not get sent to the cache.
- If the request is an ASP or CGI request, or a dynamically generated page, it does not get sent to the cache.
- If the request is a cookie, it can optionally bypass the cache.

Network administrators can configure up to 32 URL expressions, each 8 bytes long, for non-cacheable content types. Up to 128 strings (on an A180e, A184, AD3, and AD4 WebSwitch), comprising 40 bytes each, can be used for URL substring matching. As each URL Web request is examined, non-cacheable items are forwarded to the origin server and requests with substring matches are redirected to the appropriate cache server.

NOTE – The term “origin server” refers to the server originally specified in the request.

Examples of substrings are:

- “/product”: matches URL that starts with “/product,” including any information in the “/product” directory
- “product”: matches URL that has the string “product” anywhere in the entire URL

The switch is preconfigured with a list of 13 non-cacheable items that the network administrator can add, delete, or modify via the user interface. These items are either known dynamic content file extensions or dynamic URL parameters, as described below:

- dynamic content file extensions: cgi (cgi files)
- cfm (cold fusion files), .asp (ASP files), bin (bin directory), cgi-bin (cgi-bin directory),.shtml (scripted html), .htx (Microsoft HTML extension file), .exe (executable)
- dynamic URL parameters: +, !, %, =, &

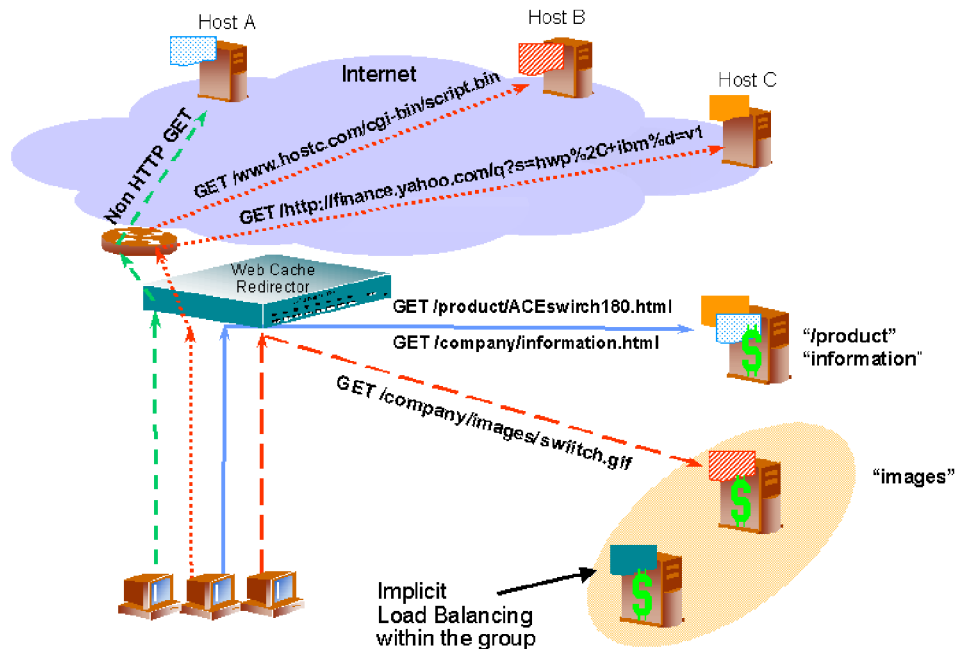


Figure 7-2 URL-Based Web-Cache Redirection

Requests will be load balanced among the multiple servers matching the URL according to the metric specified for the server group (leastconns is the default).

Configuring URL-Based Web-Cache Redirection

Web OS supports three types of NAT (Network Address Translation.) You may choose any one depending upon how you want to change the addresses for sending traffic to the Web Cache Redirection (WCR).

No NAT

In this method of NAT, the traffic is redirected to the web cache without modifying both the IP address and the MAC address of the source or origin server. The destination IP address also remains unchanged. Only the destination MAC address is changed to the MAC address of the cache. This works well for transparent cache servers. Transparent cache server processes the traffic destined to its MAC address but with the IP address of some other device.

Half NAT

In this most commonly used method of Network Address Translation, both the IP address and the MAC address of the source remain unchanged. However, the destination IP address changes to the IP address of the web cache, and the destination MAC address changes to the MAC address of the Web cache.

Full NAT

In this method of Network Address Translation, both the IP address and the MAC address of the source change to the IP address and MAC of the web cache. This method works well for proxy cache servers.

To configure URL-based WCR, perform the following steps:

- 1. Before you can configure URL-based WCR, configure the switch for basic server load balancing. 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.
 - Define each real server.

For information on how to configure your network for server load balancing, see Chapter 2.

- 2. Configure the switch to support basic WCR.**

For information on WCR, refer to Chapter 3, “Application Redirection.”

3. Configure the parameters and file extensions that will bypass WCR.

The switch is preconfigured with a list of thirteen non-cacheable items. These items are either known dynamic content file extensions or dynamic URL parameters:

- dynamic content file extensions: cgi (cgi files), .cfm (cold fusion files), .asp (ASP files), bin (bin directory), cgi-bin (cgi-bin directory), .shtml (scripted html), .htx (Microsoft HTML extension file), .exe (executable)
- dynamic URL parameters: +, !, %, =, &

a) Add or remove expressions that should not be cacheable.

```
>> # /cfg/slb/url/redirect/add|remove <expression>
```

b) Enable/disable ALLOW for none GETS (e.g., HEAD, POST, PUT) to origin server, as described below.

```
>> # /cfg/slb/url/redirect/urlal ena|dis
```

- ☐ **Enable:** Switch will allow all non-GET requests to the origin server.
- ☐ **Disable:** Switch will compare 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/url/redirect/cookie ena|dis
```

- ☐ **Enable:** Switch will redirect all requests that contain “cookie:” in the HTTP header to the origin server.
- ☐ **Disable:** Switch will compare 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/url/redir/nocache ena|dis
```

- ❑ **Enable:** Switch will redirect 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.
- ❑ **Disable:** Switch will compare 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 Web-cache SLB. Refer to the parameters listed below:

```
>> # /cfg/slb/url/lb/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 Web-cache requests. A string that starts out with a backslash (/) such as “/images” indicates that, if this string is applied to a particular server, the server can only handle requests that start out with the “/images” string.

Example: With the “/images” string, the server will handle the following requests:

```
/images/product/b.gif
/images/company/a.gif
/images/testing/c.jpg
```

This server will not handle these requests:

```
/company/images/b.gif
/product/images/c.gif
/testing/images/a.gif
```

A string that doesn't start out with a backslash (/) indicates that, if this string is applied to a particular server, the server can handle any requests that contain the defined string.

Example: With the “images” string, the server will handle 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
```

If a server is configured with only the load balance string (/), it will only handle requests to the ROOT directory.

Example: With the “(/)” string, the server will handle these requests:

```
/
/index.htm
/default.asp
/index.shtm
Any files in the ROOT directory
```

For easy configuration and identification, each defined string has an ID attached, as shown in the following example:

Example: Number of entries: six

ID	SLB String
1	any
2	.gif
3	/sales
4	/xitami
5	/manual
6	.jpg

5. Configure the real server(s) to handle WCR.

NOTE – If you don’t add a defined substring (or add the defined substring “any”), the server will handle any request.

a) To add a defined substring:

```
>> # /cfg/slb/real 2/layer7/addlb <ID>
```

Where *ID* is the identification number of the defined string.

b) To remove a defined substring:

```
>> # /cfg/slb/real 2/layer7/remlb <ID>
```

The server can have multiple defined substrings:

- “/images”
- “/sales”
- “.gif”

With these defined strings, this particular server can handle requests that start out with “/images” or “/sales” and any requests that contain “.gif”

6. On the switch, define a real server group and add real servers to the real server group.

This combines the three real servers into one server 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)
```

7. Configure a filter to support basic WCR.

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)
```


8. Enable URL-based WCR on the same filter.

The three options for configuring Network Address Translation are listed below. For more information about each of the following options, see [“Configuring URL-Based Web-Cache Redirection” on page 172.](#)

```
>> # /cfg/slb/filt <filter number>/adv/urlp ena
```

■ 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/filt <filter number>/adv/proxy ena
>> # ../rport 3128 (This port number is an example)
>> # ../port <port number>/pip 12.12.12.12 (Configure Proxy IP address on the
physical port)
>> # proxy ena (Enable the Proxy IP address)
```

9. On the switch, create a default filter for non-cached traffic.

```
>> # /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.

10. Turn on filtering for the port.

```
>> # /cfg/slb/port <port-number>/filt ena
```

11. Add the filters to the client port.

```
>> # /cfg/slb/port <port-number>/add <filter-number>
```

12. Enable Direct Access Mode (DAM) on the switch.

```
>> # /cfg/slb/adv/direct ena
```

13. On the switch, enable, apply, and verify the configuration.

```
>> SLB port <port-number># /cfg/slb                (Select the SLB Menu)
>> Server Load Balancing# on                      (Turn SLB on)
>> Server Load Balancing# apply                    (Make your changes active)
>> Server Load Balancing# cur                      (View current settings)
```

Statistics for URL-Based WCR

To show the number of hits to the cache server or origin server, use this command:

```
>> # /stats/slb/url/redir
```

Sample Statistics:

```
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
```

URL-Based Server Load Balancing

URL-based SLB allows the network administrator to optimize resource access and server tuning. Content dispersion can be optimized by basing load balancing decisions on the entire path/filename of each URL.

URL-based load balancing operates in a manner similar to URL parsing for Web-cache redirection except that the switch virtual IP (VIP) address is the target of all IP/HTTP requests.

NOTE – Both HTTP 1.0 and HTTP 1.1 requests are supported.

Network administrators can configure up to 128 strings, comprising 40 bytes each, for URL matching. Each URL Web request is then examined against the URL strings defined for each real server, as described under “[URL-Based Web Cache Redirection](#)” on page 170. URL requests will be load balanced among the multiple servers matching the URL, according to the metric specified in the server group (leastConns is the default).

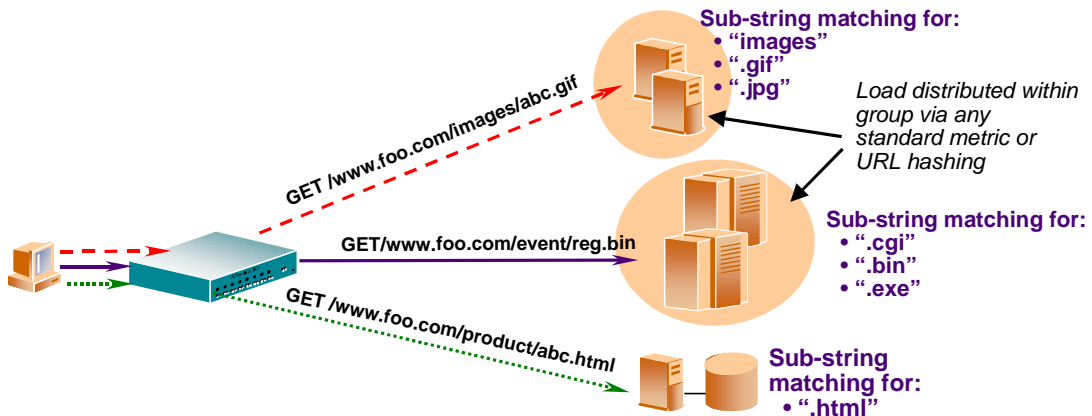


Figure 7-3 URL-Based Server Load Balancing

Example: The network administrator specifies the following criteria for load balancing:

- Requests with “.cgi” in the URL: forwarded to servers RIP1, RIP2, RIP5.
- Requests with the substring “images” in the URL: sent to servers RIP3, RIP4 and RIP6.
- Requests with URLs starting with the substring “/product:” sent to servers RIP2, RIP3 and RIP5.
- Requests containing URLs with anything else: sent to servers RIP1, RIP2, RIP3. These servers have been defined with the “any” string.

Configuring URL-Based SLB

NOTE – When URL-based SLB is used in an active/active redundant setup, use a PIP address instead of DAM to enable the URL parsing feature.

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. 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 and set up health checks for the group.
- Define a virtual server on virtual port 80 (HTTP) 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 server load balancing, see Chapter 1.

2. Define the string(s) to be used for URL load balancing. Refer to the information and examples given below:

```
>> # /cfg/slb/url/lb/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 Web-cache requests. A string that starts out with a backslash (/) such as “/images” indicates that, if this string is applied to a particular server, the server can only handle requests that start out with the “/images” string.

Example: With the “/images” string, the server will handle these requests:

```
/images/product/b.gif
/images/company/a.gif
/images/testing/c.jpg
```

This server will not handle these requests:

```
/company/images/b.gif
/product/images/c.gif
/testing/images/a.gif
```

A string that doesn't start out with a backslash (/) indicates that, if this string is applied to a particular server, the server can handle any requests that contain the defined string.

Example: With the “images” string, the server will handle 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
```

If a server is configured only with the load balance string (/), it will only handle requests to the ROOT directory.

Example: With the “(/)” string, the server will handle these requests:

```
/
/index.htm
/default.asp
/index.shtm
Any files in the ROOT directory
```

For easy configuration and identification, each defined string has an ID attached, as shown in the following example:

Example: Number of entries: six

ID	SLB String
1	any
2	.gif
3	/sales
4	/xitami
5	/manual
6	.jpg

3. Configure one or more real servers to handle URL-based load balancing.

- To add a defined substring, use this 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 substring (or add the defined substring "any"), the server will handle any request.

- To remove a defined substring, use this command:

```
>> # /cfg/slb/real 2/layer7/remlb ID
```

The server can have multiple defined substrings:

- "/images"
- "/sales"
- ".gif"

With these defined strings, this particular server can handle requests that start out with "/images" or "/sales" and any requests that contain ".gif"

4. On the switch, enable SLB.

```
>> # /cfg/slb/on (Turn SLB on)
```

5. Either enable DAM on the switch or configure a PIP address on the client port.

To use cookie-based preferential load balancing without DAM, you need to configure a proxy IP address on the client port.

NOTE – If VMA is enabled, you need to configure a PIP address on ports 1-8.

On the port you'll use for cookie-based preferential load balancing, you will enable proxy load balancing. If VMA is enabled on the switch, you can choose to configure the remaining ports with proxy IP disabled.

- To turn on DAM:

```
>> # /cfg/slb/adv/direct ena
```

- To turn off DAM and configure a PIP address on the client port:

```
>> # /cfg/slb/direct dis
>> # port 2/pip 12.12.12.12
>> # proxy ena
```

NOTE – By enabling DAM on the switch or, alternatively, disabling DAM and configuring a PIP address on the client port, port mapping for URL load balancing can be performed.

6. Enable URL-based SLB on the virtual server(s).

```
>> # /cfg/slb/virt <virtual-server-number>/service 80/httpslb ena/urlslb
```

Statistics for URL-Based SLB

To show the number of hits to the SLB or cache server, use this command:

```
>> # /stats/slb/url/lb
```

Sample Statistics:

ID	SLB String	Hits
1	any	73881
2	.gif	0
3	/sales	0
4	/xitami	162102
5	/manual	0
6	.jpg	0

HTTP Header Inspection

HTTP headers are used to include additional information to requests and responses. The HTTP 1.1 specification defines a total of 46 headers. HTTP headers can be general headers, request headers, response headers, and 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.

NOTE – One HTTP header is supported globally for the entire switch.

Customer applications of 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

Multiple Frames Processing for Delayed Binding

In addition to the URI path, which generally is less than 300 bytes, the HTTP GET requests also include general headers and request headers. HTTP cookie request headers can be 4500 bytes in length. A single GET request can include multiple cookies.

To handle the overall length of HTTP headers, including request headers containing multiple cookies, and the Maximum Segment Size (MSS) of dial-up connections, Web OS software provides the following support:

- Parsing of HTTP GET requests for URI path matching and HTTP headers matching beyond the first frame while performing delayed binding.
- Buffering of a maximum of 4,500 bytes in total for a single GET request across multiple frames.
- Processing multiple frames from a single HTTP GET request, using a TCP stack on the Switch Processor.

HTTP Header-Based SLB

By configuring HTTP header server load balancing, you can load balance HTTP requests based on different HTTP header information, such as “Cookie:” header for persistent load balancing, ‘Host:’ header for virtual hosting, and “User-Agent” for browser-smart load balancing.

NOTE – Cookie-based persistent load balancing is described in Chapter 8, “Persistence.” Virtual hosting and browser-smart load balancing is discussed in this chapter.

No Cache/Cache Control for WCR

Using this feature, you can offload 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. By enabling this feature, HTTP 1.1 GET requests with the **Cache-Control: no cache** directive in the requests are forwarded directly to the origin servers.

NOTE – For WCR, one HTTP header is supported globally for the entire switch.

In HTTP 1.0, the equivalent of the HTTP 1.1 Cache-Control: Header is the **Pragma: no-cache** header. By enabling this, requests with the **Pragma: no-cache** headers are forwarded to the origin server. This allows a client to insist upon receiving an authoritative response to its requests.

Configuring HTTP Header-Based Web-Cache Redirection

By configuring HTTP header WCR, we can redirect web-cache requests based on different HTTP header information, such as ‘Host:’ header or “User-Agent” for browser-smart load balancing.

To configure the switch to do WCR 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 Chapter 1). 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
- 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 “Host:” header.

```
>> # /cfg/slb/url/redirect/header ena host
```

4. Define the host names.

```
>> # /cfg/slb/url/lb/add ".com"
>> Server Loadbalance Resource# add ".org"
>> Server Loadbalance Resource# add ".net"
```

5. Configure the real server(s) to handle the appropriate load balance string(s).

To add a defined substring:

```
>> # /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 substring (or add the defined substring “any”), the server will handle any request.

6. If Host: header filtering is configured, you can configure the switch to use the host header field to determine whether requests are cacheable (or non-cacheable).

Example:

If you want all domain names that end with .net or .uk not to go to a cache.

a) Configure the Host header filter.

```
>> # /cfg/slb/filt 1/adv/urlp ena
```

b) Add in expression entries:

```
>> # /cfg/slb/url/redirect/add .net .uk
```

7. **You can direct a cacheable URL request to a specific cache server by configuring `min-misses` or `hash` as the metric. The switch will then use the host field in the HTTP header and the number of bytes into the URI to calculate the hash key.**

If the host field doesn't exist and no length was specified, the switch will use the source IP address as the hash key. If host field doesn't exist, but length was specified, the switch will use all or part of the URI to calculate the hash key.

```
>> # /cfg/slb/url/redirect/hash enable|disable
```

- **Enable:** Enable hashing based on the URI and set the length of URI that will be used to hash into the cache server.
- **Disable:** By disabling hashing based on the URI, the switch will only use the host header field to calculate the hash key.

Example 1. Using the source IP address as the hash key:

```
client1 requests http://www.yahoo.com --> cache1
client2 requests http://www.yahoo.com --> cache2
client3 requests http://www.yahoo.com --> cache3
```

Example 2. Using the host field and/or part or all of the URI, the same URL request will go to the same cache:

```
client1 requests http://www.yahoo.com --> cache1
client2 requests http://www.yahoo.com --> cache1
client3 requests http://www.yahoo.com --> cache1
```

Example 3. If the Host field doesn't exist, but length was specified:

```
client1 requests http://www.yahoo.com/sales/index.htm --> cache1
client2 requests http://www.yahoo.com/sales/index.htm --> cache1
client3 requests http://www.yahoo.com/sales/index.htm --> cache1
```

Configuring Browser-Based WCR

To configure User-Agent: header-based WCR, perform the following procedure.

1. **Before you can configure header-based WCR, ensure that the switch has already been configured for basic 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.
- 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/url/redirect/header enable useragent
```

4. **Define the host names.**

```
>> # /cfg/slb/url/lb/add "Mozilla"
>> Server Loadbalance Resource# add "Internet Explorer"
>> Server Loadbalance Resource# add "Netscape"
```

5. **Configure the real server(s) to handle the appropriate load balance string(s).**

NOTE – If you don't add a defined substring (or add the defined substring “any”), the server will handle any request.

To add a defined substring:

```
>> # /cfg/slb/real 2/layer7/addlb ID
```

where *ID* is the identification number of the defined string.

Virtual Hosting

Increasingly, individuals and companies are interested in having a presence on the Internet in the form of a dedicated website address. They want, for example, to 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, Web OS 8.3 enables service providers to create a single Virtual IP address to host multiple Websites per customer, each with their own hostname.

NOTE – For server load balancing, one HTTP header is supported per Virtual IP address.

The following bullets provide more detail, followed by configuration details.

- Currently, 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 is:

```
GET /products/180/ HTTP/1.0
User-agent: Mozilla/3.0
Accept: text/html, image/gif, image/jpeg
```

The GET request does not include the hostname. From the TCP/IP headers, the origin server knows its hostname, port number, and the protocol that it speaks.

- With the extension to HTTP/1.1 to include the HTTP HOST: header, the above request to retrieve the URL “/www.alteonwebsystems.com/ products/180” would look like this:

```
GET /products/180/ HTTP/1.1
Host: www.alteonwebsystems.com
User-agent: Mozilla/3.0
Accept: text/html, image/gif, image/jpeg
```

The Host: header carries the hostname used to yield the IP address of the site.

- Based on the Host: header, the switch will forward the request to servers representing different customers’ websites.
- Network administrator needs to define a domain name as part of the 128 supported URL substrings.
- The switch will perform substring matching; that is, the substring “alteonweb-systems.com” or “www.alteonwebsystems.com” will match “www.alteonweb-systems.com.”

Virtual Hosting Configuration Overview

The sequence of events for configuring virtual hosting, based on HTTP Host: headers, is described below:

1. **Network administrator defines a domain name as part of the 128 supported URL substrings.**

Both domain names “www.company-a.com” and “www.company-b.com” get resolved to the same IP address. In this example, the IP address is a VIP address on the switch.

2. **“www.company-a.com” and “www.company-b.com” are defined as URL substrings.**

3. **Server Group 1 is configured with Servers 1 – 8.**

Servers 1 – 4 belong to “www.company-a.com” and Servers 5 – 8 belong to “www.company-b.com.”

4. **Network administrator assigns substring “www.company-a.com” to Servers 1 – 4 and substring “www.company-b.com” to Servers 5 – 8.**

5. **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 – 4.
- If the host header is “www.company-b.com,” the switch directs requests to one of the Servers 5 – 8.

Configuring the “Host:” Header for Virtual Hosting

To configure “Host:” header load balancing to support virtual hosting, perform the following procedure:

1. **Before you can configure header-based load balancing, ensure that the switch has already been configured for basic 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.
- Assign servers to real server groups.
- Define virtual servers and services

For information on how to configure your network for server load balancing, see Chapter 1.

2. **Turn on URL parsing to the virtual server for virtual hosting.**

```
>> # /cfg/slb/virt 1/service 80/httpslb ena host
```

3. **Define the host names.**

```
>> # /cfg/slb/url/lb/add "www.customer1.com"
>> Server Loadbalance Resource# add "www.customer2.com"
>> Server Loadbalance Resource# add "www.customer3.com"
```

4. **Configure the real server(s) to handle the appropriate load balance string(s).**

To add a defined substring:

```
>> /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 substring (or add the defined substring “any”), the server will handle any request.

Browser-Smart Load Balancing

By inspecting the “User-Agent” header, requests can be directed to different servers based on browser type.

Configuring Browser-Based Load Balancing

To configure “User-Agent:” header load balancing to allow the switch to perform browser-smart 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. 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.
 - Assign servers to real server groups.
 - Define virtual servers and services.
2. **Turn on URL parsing to the virtual server for “User-Agent:” header.**

```
>> # /cfg/slb/virt 1/service 80/httpslb ena browser
```

3. **Define the host names.**

```
>> # /cfg/slb/url/lb/add "Mozilla"
>> Server Loadbalance Resource# add "Internet Explorer"
>> Server Loadbalance Resource# add "Netscape"
```

4. **Configure the real server(s) to handle the appropriate load balance string(s).**

NOTE – If you don’t add a defined substring (or add the defined substring “any”), the server will handle any request.

To add a defined substring:

```
>> # /cfg/slb/real 2/layer7/addlb ID
```

where *ID* is the identification number of the defined string.

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.

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 website
- Provide better services to repeat customers, based on access count

Clients to 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, log-in 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, log-in authentication, or permanent HTTP cookie.

- Applications

All users using a specific application. For example, giving priority to HTTPS traffic that is performing credit card transactions versus HTTP browsing traffic.

- Content

Users accessing specific content.

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. 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.
- Assign servers to real server groups.
- Define virtual servers and services.

For information on how to configure your network for server load balancing, see Chapter 1.

2. Turn on URL parsing to the virtual server.

```
>> # /cfg/slb/virt 1/service 80/httpslb 80 enable cookie sid 1 6 dis
```

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)

3. Define the cookie values.

```
>> # /cfg/slb/url/lb/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's 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.

4. Configure the real server(s) to handle the appropriate load balance string(s).

To add a defined substring:

```
>> # /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 substring (or add the defined substring "any"), the server will handle any request.

URL Hashing

By default, hashing algorithms use the IP source address and/or IP destination address, depending on the application area, to determine content location. For example, Firewall Load Balancing uses both IP source and destination addresses, Web-Cache Redirection uses only the IP destination address, and server load balancing uses only the IP source address.

If URL-based WCR is enabled and the “Host”: header is present in the URL of an HTTP request, you can hash on the header or the URL to determine content location. All requests for “*www.alteonwebsystems.com*,” for example, will be forwarded to the same cache server. By default, URL hashing is disabled. When enabling this option, the network administrator must also specify the number of bytes (up to 255) to be used for hashing the URL.

The applications of URL hashing for WCR and SLB are described below.

URL Hashing for Web-Cache Redirection

Using the hashing algorithm, you can optimize “cache hits,” redirecting client requests going to the same page of an origin server to a specific cache server.

- The load-balancing algorithm must be configured to be “hash” or “minmiss.”
- Hashing is based on the URL, including the HTTP Host header (if present), up to a maximum of 255 bytes.

Example: The switch will use the string “*alteonwebsystems.com/products/180*” for hashing the following request:

```
GET http://products/180 / HTTP/1.0
```

```
HOST:www.alteonwebsystems.com
```

URL Hashing for Server Load Balancing

The default hashing algorithm for VIP load balancing is the IP source address. By enabling URL hashing, requests going to the same page of an origin server will be redirected to the same real server (RIP) or cache server.

- The load-balancing algorithm must be configured to be “hash” or “minmiss.”
- Hashing is based on the URL, including the HTTP Host: header (if present), up to a maximum of 255 bytes.

VIP Load Balancing of Non-Transparent Caches

Customers can deploy a cluster of non-transparent proxy caches and use the VIP method to load balance requests to these cache servers.

The client's browser will be configured to send Web requests to a non-transparent cache (the IP address of the VIP configured).

If hash is selected as the load-balancing algorithm, the current hashing algorithm will only use the IP Source Address for hashing in SLB. Thus, the switch may not send Web requests for the same origin server to the same proxy cache server. For example, requests made from a client to “http://www.alteonwebsystems.com/products” from different clients may get sent to different caches.

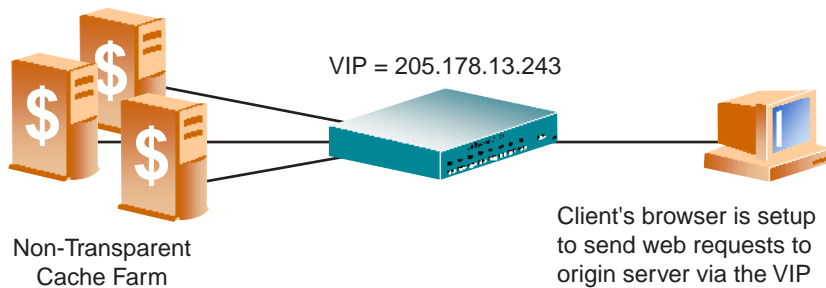


Figure 7-4 Balancing Non-Transparent Caches

Configuring URL Hashing

You can direct the same URL request to the same cache or proxy server that uses a VIP address to load balance proxy requests. By configuring hash or minmisses as the metric, the switch will use the number of bytes into 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 will also use 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. 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.
- Assign servers to real server groups.
- Define virtual servers and services.

For information on how to configure your network for server load balancing, see Chapter 1.

2. Enable URL parsing.

```
>> # /cfg/slb/virt 1/service 80/httpslb enable urlhash 25
```

3. Set the metric for the real server group to minmisses or hash.

```
>> # /cfg/slb/group 1/metric hash|minmiss
```

Exclusionary String Matching for URL SLB

URL-based SLB and WCR can match up to 128 substrings.

Examples of substrings are

- “/product,” matches URL that starts with “/product”
- “product,” matches URL that has the string “product” anywhere in the entire URL

You can assign one or more substring to real servers. When more than one URL substring is assigned to a real server, requests matching any substring will be redirected to that real server. There is also a special substring known as “any” that matches all content.

Web OS supports exclusionary substring matching. Using this option, an administrator can define a server to accept any requests regardless of the URL, except requests with a specific substring. That is, an administrator can define the URL substring to be excluded, assign it to a real server, and have the server interpret it as an exclusion instead of an inclusion.

NOTE – Once exclusionary substring 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, while at the same time have it exclude other URL strings. The exclusionary feature is enabled per server, not per string.

Example:

substring #1 = cgi
substring #2 = NOT cgi/form_A
substring #3 = NOT cgi/form_B

When these substrings are assigned to a real server, the behavior is to match all cgi scripts, but exclude form_A and form_B.

Configuring Exclusionary URL Substring Matching

To configure exclusionary URL substring matching, perform the following procedure:

1. Before you can configure URL substring matching, ensure that the switch has already been configured for basic 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.
- Assign servers to real server groups.
- Define virtual servers and services.

For information on how to configure your network for server load balancing, see Chapter 1.

By default, this feature is disabled. In order to enable it, you must add at least one load balancing string to the server.

```
>> # /cfg/slb/real 1/exclude enable|disable
```

Example 1:

```
205.178.15.49, enabled, name, weight 1, tmout 10, maxcon 200000
backup none, inter 10, retry 4, restr 8, remote disabled, proxy
enabled
handle URL cookie: disabled
exclusionary string matching: enabled
2: test
real ports:
  http: vport http, group 1, httpslb
  URL hashing: disabled
  virtual server: 1, 205.178.15.45, enabled
```

This server will handle any requests except requests containing the string “test.”

Example 2:

```

205.178.15.49, enabled, name, weight 1, tmout 10, maxcon 200000
backup none, inter 10, retry 4, restr 8, remote disabled, proxy
enabled
handle URL cookie: disabled
exclusionary string matching: enabled

2: test
3: /images
4: /product
real ports:
  http: vport http, group 1, https1b
  URL hashing: disabled
  virtual server: 1, 205.178.15.45, enabled

```

This server will handle any requests EXCEPT requests contain the string “test” OR requests that start with “/images” OR request start with “/product.”

Example 3:

```

205.178.15.49, enabled, name, weight 1, tmout 10, maxcon 200000
backup none, inter 10, retry 4, restr 8, remote disabled, proxy
enabled
handle URL cookie: disabled
exclusionary string matching: enabled

1: any
real ports:
  http: vport http, group 1, https1b
  URL hashing: disabled
  virtual server: 1, 205.178.15.45, enabled

```

This server will not handle ANY requests, which is the same as disabling this server!



CHAPTER 8

Persistence

Session persistence allows you to re-establish a user's connection to a particular server. This 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.

Persistence-based load balancing enables the network administrator to redirect requests from a client to the real server that initially handled the request. Persistence can be based on IP source address, HTTP cookies for HTTP requests, or SSL session ID for encrypted HTTPS requests.

IP Source Address-Based Persistence

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 problems associated with session persistence based on a packet's IP source address:

- **No Server Load Balancing (SLB):** Proxied clients will appear to the switch as a single IP source address and will not be able to take advantage of server load balancing on the switch. When many individual users behind a firewall use the same IP proxy source address, requests will be 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.
- **No Persistence:** When individual users share a pool of IP source addresses, persistence for any given request cannot be assured. Although each IP source address will be directed to a specific server, the address itself is randomly selected, thereby making it impossible to predict which server will receive the request. SLB is supported, without true persistence for any given user.

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 need to use 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 basket 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 allow their pages to be customized.

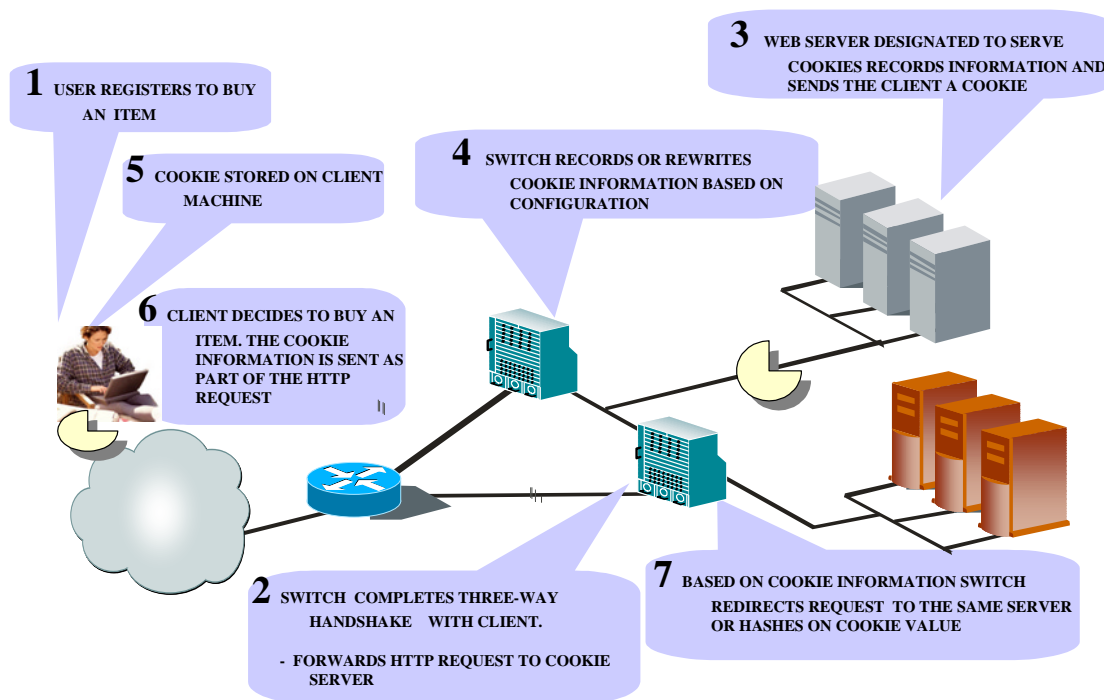


Figure 8-1 Cookie-Based Persistence: How It Works

Types of HTTP Cookies

A cookie can be defined in the HTTP header (the recommended method) or placed in the URL for hashing. On a switch running Web OS, you configure cookie-based persistence with the cookie name, offset, length, and where to match this cookie value (in the cookie header or the URL). The default is to match the cookie in the cookie header.

Web OS provides the following support:

- Cookie names of up to 20 bytes.
- Cookie values of up to 64 bytes for hashing.
This is applicable only for the passive cookie mode, using a temporary cookie. The switch hashes the cookie value to determine which server to forward the request to.
- An asterisk (*) is supported in cookie names for wildcards.
For example, Cookie name = ASPsession*

The format of a cookie defined as an HTTP header is a "Name=Value" pair, in addition to other parameters. For example, the cookie "SessionID=1234" can be represented by the following:

- Cookie Header
Cookie:SessionID=1234
- Cookie within the URL
/www.travelocity.com/Reservation/SessionID=1234

Cookies can either be permanent or temporary. A *permanent cookie* gets stored on the client's browser, as part of the response from a 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 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.

NOTE – If you will be using temporary cookies, the passive cookie mode is recommended.

Examples of cookies are given below:

Cookie: ASP_SESSIONID=POIUHKJHLKHD

Cookie: name=john_smith

The first example represents an Active Server Page (ASP) session ID. The second example represents an application-specific cookie that records the name of the client.

Modes of Operation

There are two cookie modes used to maintain session persistence; *passive* and *active (cookie rewrite)*. Passive cookie mode works for both cookies defined in the HTTP cookie header and cookies defined in the URL. Active cookie mode (cookie rewrite mode) can only be used with cookies defined in the HTTP cookie header.

Passive Cookie Mode

In this mode, there is no special persistence cookie defined on the server. The network administrator configures the Web server to embed a cookie in the server response that the switch looks for in subsequent requests from the same client. This is the recommended mode of operation when using temporary cookies.

NOTE – Passive cookie mode is not compatible with Global Server Load Balancing (GSLB). A customer running GSLB who needs cookie-based persistence should use active cookie mode for maintaining persistence.

The following figure shows passive cookie mode operation.

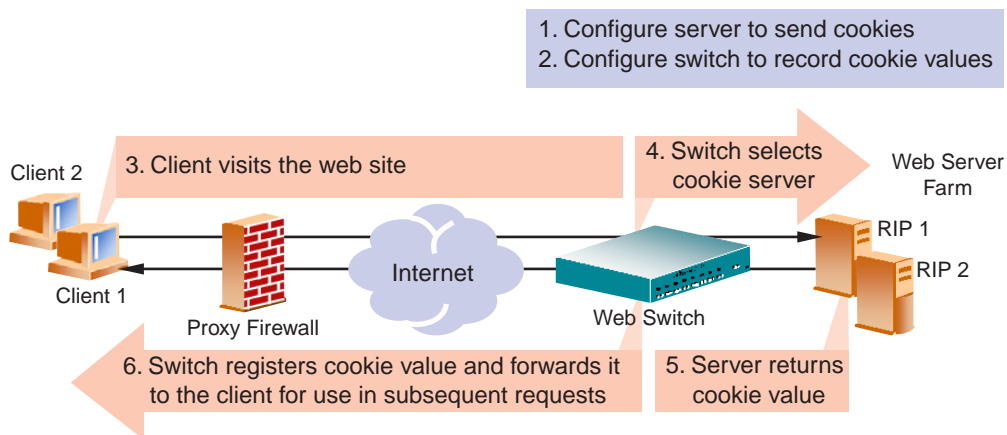


Figure 8-2 Passive Cookie Mode

Subsequent requests from Client 1 with the same cookie value will be sent to the same server (RIP 1 in this example).

Active Cookie Mode (Cookie Rewrite Mode)

In active cookie mode (or cookie rewrite mode), the switch generates the cookie value on behalf of the server, eliminating the need for a network administrator to generate cookies for each user. The server is configured to return a special persistence cookie that is predefined on the switch and on the server. The switch then intercepts this persistence cookie and rewrites the value to include server-specific information before sending it on to the client.

- Active cookie mode requires at least 8 bytes in the cookie header. An additional 8 bytes must be reserved if you are using cookie-based persistence with GSLB.

NOTE – Active cookie mode (cookie rewrite mode) only works for cookies defined in the HTTP cookie header, not cookies defined in the URI. The switch only rewrites the cookie in the first request in a TCP session. Care should be taken when deploying active cookie mode with HTTP 1.1 because multiple HTTP GET requests happen within the same TCP connection. To support the active cookie mode correctly in HTTP 1.1, you need to ensure that the application issues the persistence cookie in the first HTTP GET to the site.

Example: The following figure shows active cookie mode operation:

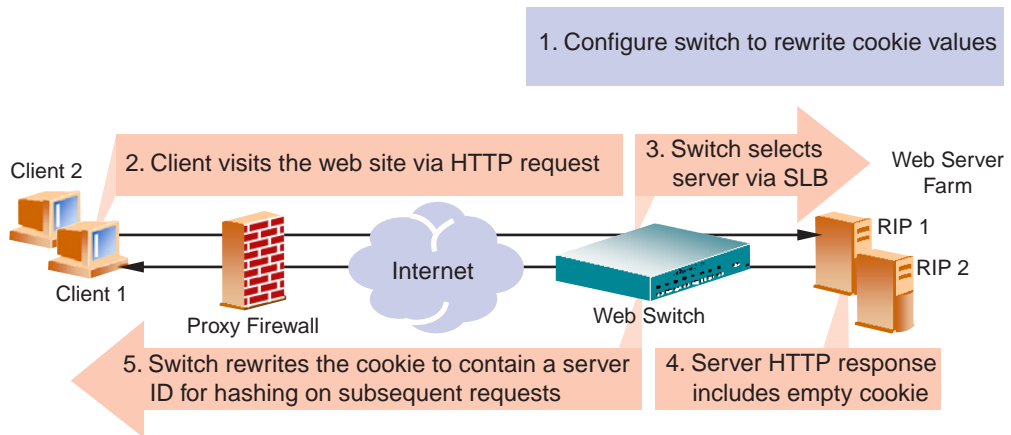


Figure 8-3 Active Cookie Mode

NOTE – When the 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 server ID or it can be the server IP address or server ID. The rewritten cookie value is encoded. Subsequent requests from Client #1 with the same cookie value will be sent to the same server.

Cookie Assignment Servers

Servers defined within a group can be configured to assign/issue cookies to first-time visitors to the website. These are known as *cookie assignment servers*. If there are multiple cookie assignment servers, the switch will load balance the request to these servers. Cookie assignment servers do not participate in load balancing requests that already have a cookie assigned.

Cookie assignment servers are not required for cookie-based persistence. If there are no cookie assignment servers defined, the switch assumes that either the client's request contains a cookie or that the real server that gets the first request without a cookie will issue a cookie value back to the client browser.

Cookie Values

Cookie assignment servers or any real server can return a cookie value to the client browser. The entire cookie value or part of it is used for selecting the appropriate server for directing the request. Two examples of cookies are shown below:

Example 1: `cookie: sid=1234cdb20f043243`

Example 2: `cookie: sid=0123456789abcdef`

In the first example, the server's IP address has been embedded in the cookie value returned by the server. The value "cdb20f04" represents the IP address "205.178.15.4" in hexadecimal format. In this example, the user has defined the following for use to compute the server to redirect the request:

Cookie name = sid

Offset = 5 (the starting position of the value to be used for hashing)

Length = 8 (the number of bytes from the starting position)

Since the defined value represents a valid real server IP address, the switch will use the value directly to determine the server that will receive subsequent requests instead of using the value for hashing.

In the second example, the defined cookie value does not match a valid real server IP address and thus will be used for hashing to determine the appropriate real server to which subsequent requests should be directed.

Sequence of Events Using Cookie Assignment Server

Two scenarios for cookie-based persistence, demonstrating use of cookie assignment servers, are given below.

NOTE – When using cookie assignment servers, you must use hash or minmisses as the load-balancing metric for the real server group.

First time a HTTP client request arrives at the switch, without the specified cookie:

- With cookie assignment servers defined:
 - The switch will forward the requests to one of the cookie assignment servers to get a cookie value return to the client browser.
 - Subsequent requests from this client with this same cookie value will get hashed to an appropriate real server based on the cookie value.
- Without cookie assignment servers defined:
 - The switch will direct the request to a real server in the group, based on the load-balancing algorithm defined.
 - Subsequent requests from this client will get directed to the same server. The cookie value can be the real server IP address or it can be a cookie value previously embedded in the server response to the client which the switch will then hash to determine the server that should receive the request.

Subsequent client HTTP requests arrive at the switch, with the specified cookie:

1. Client HTTP request sent to the switch, with the specified cookie.
2. The switch will use the “offset” and “length” parameters to determine which part of the cookie value should be used for determining to which real server to direct the request. For the discussion below, let's call this the `persistence_cookie_value`.
3. If the `persistence_cookie_value` represents a valid real server IP address, the switch will redirect the request to the appropriate real server.
4. If the `persistence_cookie_value` does not represent a valid real server IP address, the switch will use the value to compute (hash) the server that should get this request.

The concept of cookie was extended by matching cookies either in the URI or the cookie header; that is, the user can configure cookie based persistence with the cookie name, offset, length and where to match this cookie value (in the URI or the cookie header itself). This is a configuration option as described below. The default is to match the cookie in the cookie header.

Assigning Server to Serve Cookies When Client Requests Don't Contain the Specified Cookie

With the `nocookie` option enabled for specific servers in a real server group, connection requests without cookies are load balanced across those real servers. Requests with specified cookies will be load balanced across the other real servers in the group.

To assign a server specifically to serve cookies when client requests don't contain the specified cookie, use this command:

```
>> # /cfg/slb/real 1/nocook ena (Enable SLB for non-cookie requests)
```

Example:

Real Server Group 1 consists of real servers 1,2,3,4,5,6. Only real servers 5 and 6 have `/nocook` enabled.

When client requests come in for the first time and don't have the specified cookie, servers 5 and 6 will be load balanced to assign a specified cookie to the server. Subsequently, when a request from the same client comes in with the specified cookie, it will get hashed and load balanced across servers 1, 2, 3, or 4.

Using Cookie Assignment Servers: Configuration Examples

If you want to look for cookie name/value pair in the HTTP cookie header, configure the `Look for cookie in URI [e|d]:` option to be “*disable*,” as shown below:

```
>> Virtual Server 1 http Service# pbind
Current persistent binding mode: disabled
New persistent binding mode: disabled
Enter clientip|cookie|sslid|disable persistence mode:      cookie
Enter passive|rewrite cookie persistence mode [p/r]:      p
Enter Cookie Name: sid
Enter the starting point of the cookie value:      8
Enter the number of bytes to be extract:      4
Look for cookie in URI [e|d]:      d
Current persistent binding for http: disabled
New persistent binding for http: cookie
```

If you want to look for cookie name/value pair in the URI, configure the `Look for cookie in URI [e|d]:` option to be “*enable*.”

Example 1:

HTTP Header:

```
GET /product/switch/UID=12345678;ck=1234...
Host: www.alteonwebsystems.com
Cookie: UID=87654321;
....
```

- If we configure the enable | disable parameter in **/cfg/slb/virt 1/service 80/pbind cookie passive *cookie name offset length* disable** to be “disable”, the switch will look for the name/value pair in:
Cookie: UID=87654321;
- If we configure the enable | disable parameter in **/cfg/slb/virt 1/service 80/pbind cookie passive *cookie name offset length* disable** to be “enable”, the switch will look for the name/value pair in:
/product/switch/UID=12345678;ck=1234..

Example 2:

HTTP Header:

```
Cookie: sid=0123456789abcdef; name1=value1;...
```

- Use the cookie name “sid” and “789a” of the cookie value as a hashing key to compute the real server. The configuration would be:
/cfg/slb/virt 1/service 80/pbind cookie passive sid 8 4 disable

Based on the command parameters used above, the switch will use an offset of 8 bytes and a length of 4 bytes from that offset to determine which part of the cookie value should be used for determining the real server that should get the request.
- To include the entire value of “sid”, the configuration would be:
/cfg/slb/virt 1/service 80/pbind cookie passive sid 1 16 disable

Configuring Cookie-Based Persistence

When to Use Passive or Active Mode

- **Temporary cookie:** When implementing cookie-based persistence using temporary cookies, passive mode is recommended. Use `leastconns` or `roundrobin` as the load-balancing metric for the real server group. Using this configuration, no cookie assignment server(s) would be needed.
- **Permanent cookie:** Use the passive mode, with the server embedding the IP address, or use active mode.

What LB Algorithm Should I Use with Cookie-Based Persistence?

To ensure optimal load balancing, use either `leastconns` or `roundrobin` as the load-balancing metric. However, if you will be using cookie assignment servers, you must use either `hash` or `minmisses` as the load balancing metric.

Using Cookie-Based Persistence with GSLB

If you will be using cookie-based persistence with GSLB, you must use the active cookie mode and reserve 16 bytes in the cookie header.

What If Client Browser Doesn't 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 persistence enabled, session persistence for requests from a browser that doesn't accept cookies will be based on the source IP address. Many individual users coming from a proxy firewall will be directed to a single server, resulting in traffic being concentrated on a single server instead of load balanced across the available real servers.

Configuration Procedure

1. **Before you can configure cookie-based persistence, you need to configure the switch for basic server load balancing. 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, etc., as appropriate.
- Assign servers to real server groups.
- Define virtual servers and services.

For information on how to configure your network for SLB, see Chapter 2.

2. **Either enable Direct Access Mode (DAM) for the switch or disable DAM and specify proxy IP (PIP) address(es) on the client port(s).**

- Enable DAM for the switch.

```
>> # /cfg/slb/adv/direct ena (Enable Direct Access Mode on switch)
```

- Disable DAM and specify PIP address(es) on the client port(s).

NOTE – If Virtual Matrix Architecture (VMA) is enabled on the switch, you must configure a unique PIP address for every port.

```
>> # /cfg/slb/adv/direct disable (Disable DAM on the switch)
>> # /cfg/slb/port 1 (Select network port #1)
>> # pip 200.200.200.68 (Set proxy IP address for port #1)
```

3. **If PIP addresses are used, make sure server processing is disabled on the server port.**

```
>> # /cfg/slb/port 1 (Select switch port #1)
>> # server dis (Disable server processing on port #1)
```

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.

```
>> # /cfg/slb/virt 1/service 80
>> Virtual Server 1 http Service# pbind
Enter clientip|cookie|sslid persistence mode:    cookie
Enter passive|rewrite cookie persistence mode [p/r]:    passive
Enter Cookie Name: sid
Enter the starting point of the cookie value:    1
Enter the number of bytes to be extract:        8
Look for cookie in URI [e|d]:    dis
```

Once you specify “cookie” as the mode of persistence, you will be prompted for the following parameters:

- Cookie persistence mode: `passive` or `rewrite` (active)
- Cookie name
- Starting point of the cookie value
- Number of bytes to be extracted
- Enable/disable looking for cookie in the URI

NOTE – Cookie `rewrite` mode only works with cookies defined in the HTTP cookie header. When the cookie `rewrite` mode is selected, you will not be prompted for a value in this field.

To configure the switch to look for cookie name/value pair in the `Cookie:` field of the HTTP header, set the `Look for cookie in URI` option to `disable`. If you want the switch to look for the cookie name/value pair in the URI, set this option to `enable`.

Example 1:

HTTP Header:

```
GET /product/switch/UID=12345678;ck=1234...
Host: www.alteonwebsystems.com
Cookie: UID=87654321;
```

- If you set the last parameter to **disable**, the switch will look for the name/value pair in
Cookie: UID=87654321;

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive UID 1 8 dis
```

- If you set the last parameter to **enable**, the switch will look for the name/value pair in
product/switch/UID=12345678;ck=1234..

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive UID 1 8 ena
```

Example 2:

HTTP Header:

Cookie: sid=0123456789abcdef; name1=value1; ...

If you want the switch to use cookie name “sid” and “789a” of the value as a hashing key to the real server, use this command:

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive sid 8 4 dis
```

To use the whole value of “sid” as a hashing key to the real server, use this command:

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive sid 1 16 dis
```

You can also use wild cards in configuring cookie names for cookie-persistent load balancing, as shown in this command:

```
>> # /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, ASPSESSIONID789 will be seen by the switch as the same cookie name.

Directing Cookie Client to a Specific Server

This can be done using passive cookie mode or active cookie mode (cookie rewrite mode). The procedure for each mode is provided below.

Passive Cookie Mode

By embedding the real server's IP address (in hexadecimal format) in the cookie value, the cookie assignment server can tell the client exactly which server to return to in subsequent connections.

Example 1:

1. **Convert the client IP address from dotted format into hexadecimal:**

205.178.15.4 --> cdb20f04

2. **Embed the converted address into the cookie value.**

sid=1234cdb20f043243

3. **Configure the switch to read the correct cookie value.**

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive sid 5 8 dis
```

Example 2:

1. **Convert the client IP address from dotted format into hexadecimal:**

205.178.15.9 --> cdb20f09

2. **Embed the converted address into the cookie value.**

sid=cdb20f09

3. **Configure the switch to read the correct cookie value.**

```
>> # /cfg/slb/virt 1/service 80/pbind cookie passive sid 1 8 dis
```

Example 3:

Using cookie passive mode, the switch will examine the server's "Set-Cookie:" value and direct all subsequent connections to the server that assigned the cookie.

Server 1 --> Set-Cookie: sid=1234567

Client 2 --> Cookie: sid=1234567

All of Client 2's traffic with cookie sid=1234567 will be directed to real server 1.

Active Cookie Mode (Cookie Rewrite Mode)

Example 1:

If the switch is configured to be in cookie rewrite mode with the seventh parameter (byte length) configured to be 8 or 16, the switch will rewrite the cookie value with the encrypted real server IP address (RIP) or encrypted virtual server IP address (VIP) and RIP.

Cookie-based persistence is configured on the switch as follows:

```
>> # /cfg/slb/virt 1/service 80/pbind cookie rewrite sid 1 8 dis
```

Server 1 (205.178.15.4) --> Set-Cookie: sid=alteonpersistence;

The switch will rewrite: --> Set-Cookie: sid=cdb20f04rsistence;

Client --> Server 1 --> Cookie: sid=cdb20f04rsistence;

Example 2:

Cookie-based persistence is configured on the switch as follows:

```
>> # /cfg/slb/virt 1/service 80/pbind cookie rewrite sid 1 16 dis
```

VIP --> (205.178.15.10)

Server 1 (205.178.15.4) --> Set-Cookie: sid=alteonpersistence;

The switch will rewrite: --> Set-Cookie: sid=cdb20f04cdb20f0ae;

Client --> Server 1 --> Cookie: sid=cdb20f04cdb20f0ae;

SSL Session ID-Based Persistence

Secure Sockets Layer (SSL) is a set of protocols built on top of TCP/IP that allow an application server and user to communicate over an encrypted HTTP session, providing authentication, non-repudiation, and security. The SSL protocol “handshake” is performed using clear 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 request to the real server that it bound the user to during the last session. Because SSL protocol allows many TCP connections from the same client to a server to use the same session ID, key exchange needs to be done only when the session ID expires. This cuts down on CPU overhead on the server and provides a mechanism, even when the client IP address changes, to send all sessions to the same 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 is only “visible” to the switch after the TCP 3-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, or minmisses).
- 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. When a subsequent connection comes in with the same session ID, it is 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 handshake to happen.
- Session IDs are kept in the switch until an idle time equal to the configured server timeout (default = 10 minutes) for the selected real server has expired.

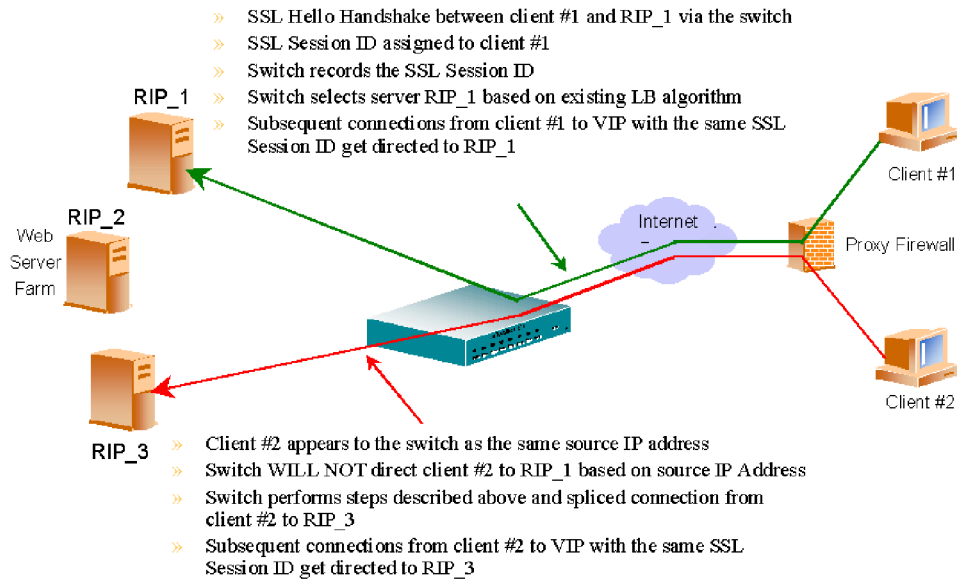


Figure 8-4 SSL Session ID-Based Persistence

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, 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 Chapter 1, “Server Load Balancing.”

2. If a proxy IP address is not configured on the client port, enable DAM for real servers.

```
>> # /cfg/slb/adv/direct ena (Enable DAM on switch)
```

3. Select the persistent binding type for the virtual port to configure session ID-based persistence.

```
>> # /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
```

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 WebSystems 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, Web OS 8.3 allows the network administrator to configure the switch to look through multiple HTTP responses from the server.

In Web OS 8.3, 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]:
```


CHAPTER 9

Bandwidth Management

Bandwidth Management (BWM) enables Website managers to allocate a certain 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.

Overview

To manage bandwidth, create one or more bandwidth management contracts. The switch uses these contracts to limit individual traffic flows.

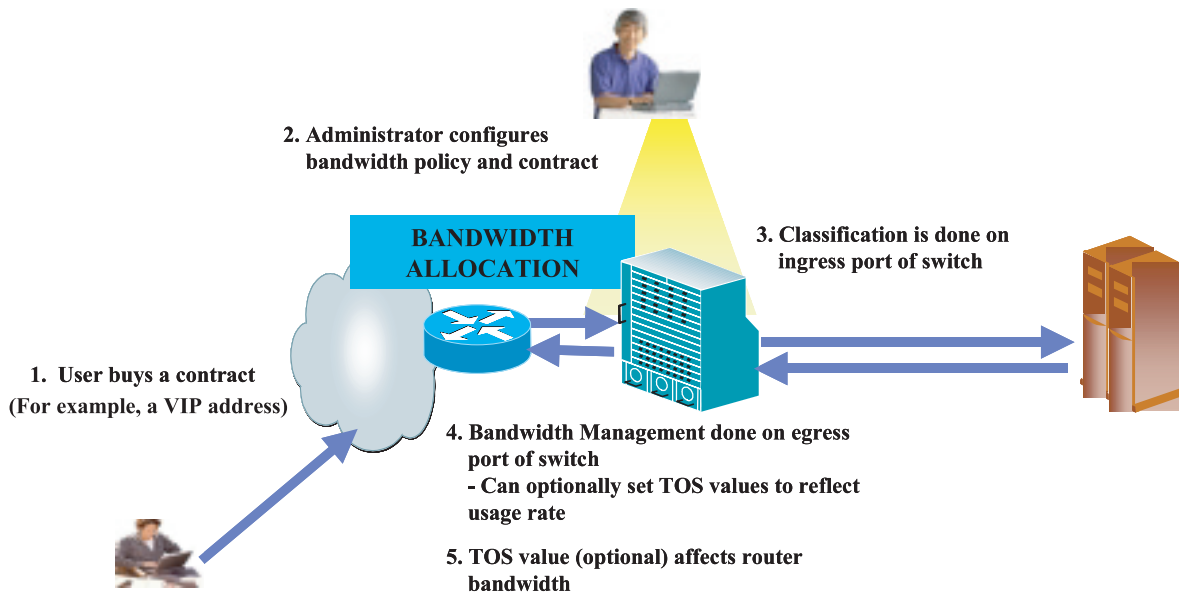


Figure 9-1 Bandwidth Management: How It Works

Each contract comprises the following:

- A classification policy where certain frames are grouped together
- A bandwidth policy specifying usage limitations to be applied to these frames

NOTE – At any given time, up to 256 BWM contracts can be configured for a single Alteon AD3 or Alteon 180e WebSwitch. Up to 1024 contracts can be created for a single Alteon AD4 or Alteon 184 WebSwitch.

- When Virtual Matrix Architecture (VMA) is not enabled, *bandwidth classification* is done on the *ingress* side of the switch (at the ingress port or designated port) and can be based on the following: source port, VLAN, filters, Virtual Internet Protocol (VIP) address, service on the Virtual server, URL, and so on.

When VMA is enabled, traffic classification that is not based on filters or SLB 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 filter-based or SLB traffic, then the classification occurs on the *designated port*.

NOTE – VMA is recommended when Bandwidth Management is enabled.

- 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 live).

Rate management 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.

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 hoster 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.

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).

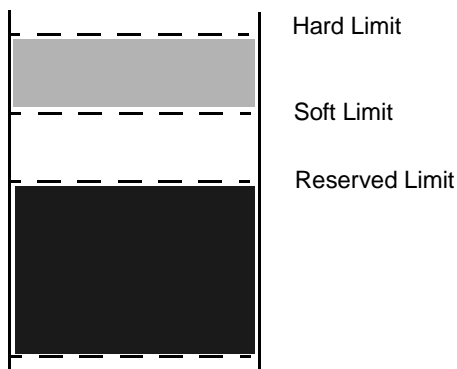


Figure 9-2 Bandwidth Rate Limits

Rate Limits

A bandwidth policy specifies three limits, listed and described in [Table 9-1](#):

Table 9-1 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 the case where the total CIRs exceed the out-bound port bandwidth, the switch will perform a graceful degradation of all traffic on the associated ports.
Soft Limit	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.
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 (or nk).

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.”

[Table 9-2](#) lists the granularity of policy limits:

Table 9-2 Bandwidth Policy Limits

Bandwidth Range	Interval	Bandwidth Range	Interval
250 Kbps to 5000 Kbps	250 Kbps	50 Mbps to 150 Mbps	10 Kbps
1 Mbps to 20 Mbps	1 Mbps	150 Mbps to 500 Mbps	25 Mbps
20 Mbps to 50 Mbps	5 Mbps	500 Mbps to 1000 Mbps	50 Mbps

In addition, a queue size is associated with each policy. The queue size is measured in bytes.

Data Pacing

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.

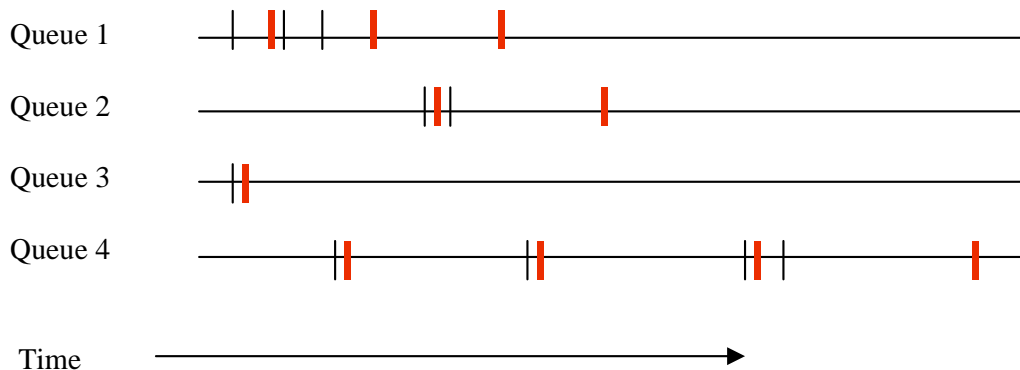


Figure 9-3 Virtual Clocks and TDT

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 Committed Information Rate (CIR) is hit. If the CIR is over-committed 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.

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.

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 229](#)).

- When both filter TOS and BWM TOS are applied, BWM TOS has precedence.
- BWM configurations will optionally be synchronized during VRRP synchronization except port contracts and VLAN contracts, which will not be synchronized.

Classification Criteria

The frames associated with a particular BWM contract are specified, using the parameters listed below. All of these classifications are aimed at limiting the traffic outbound from the server farm for bandwidth measurement and control.

Server Output Bandwidth Control

- Physical Port - All frames are from a specified physical port.
- VLAN - All frames are from a specified VLAN. Even 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 VIP.

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 particular TCP port number (either source or destination)
- UDP - All UDP frames
- UDP Port Number - All frames with a particular UDP port number (either source or destination)

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 this—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**

Bandwidth Classification Configuration

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/url/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.

Frame Discard

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 will be discarded.

URL-Based Bandwidth Management

URL-based BWM allows the network administrator or Website 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 7, “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 Website 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 9-4](#), if a Website has 10Mbs 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 WebSwitch 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 9-5 on page 231](#), users will be able to allocate a certain percentage of bandwidth for Web cache requests by using the URL parsing and bandwidth management feature.

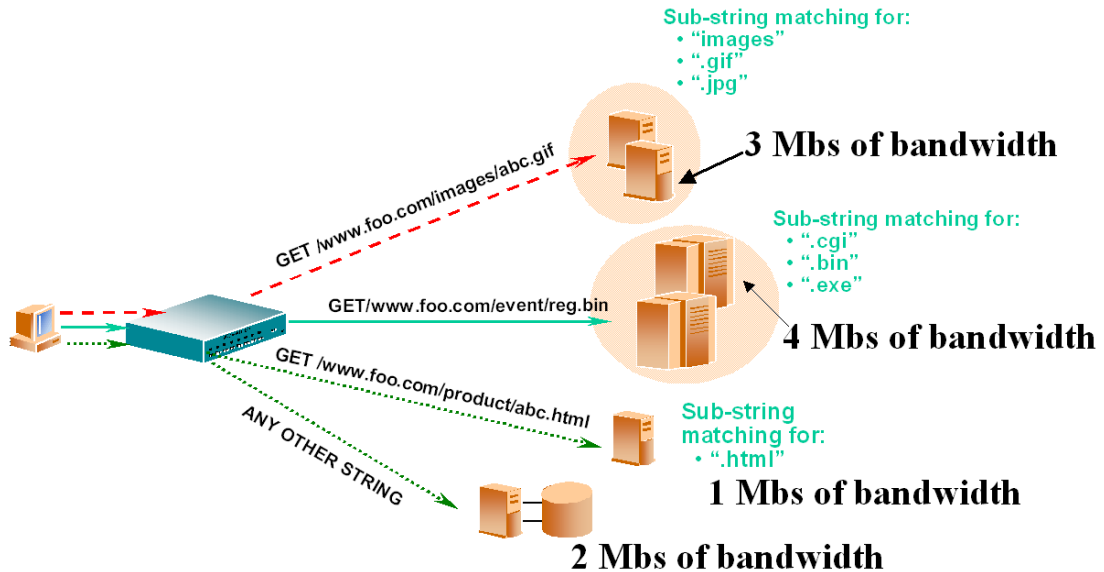


Figure 9-4 URL-Based Bandwidth Management

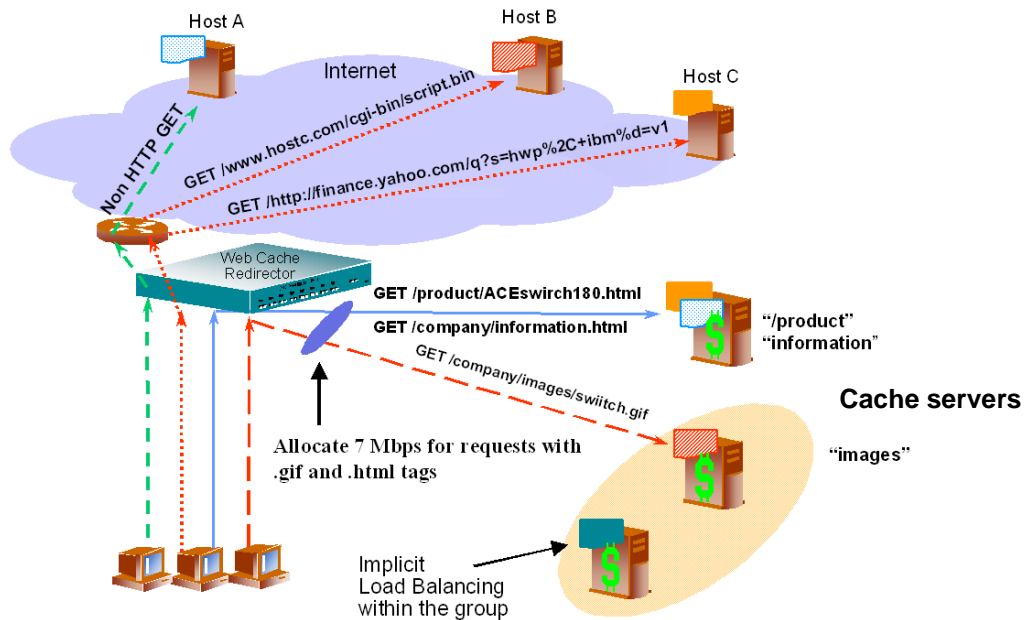


Figure 9-5 URL-Based Bandwidth Management with Web Cache Redirection

HTTP Header-Based Bandwidth Management

HTTP header-based BWM allows Website managers to allocate bandwidth based on header value. Thus, they can allocate bandwidth based on browser type, cookie value, etc.

Cookie-Based Bandwidth Management

Cookie-based BWM enables Website 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, Website managers can classify users as first class, business class and coach, and allocate a larger share of the bandwidth for preferred classes.

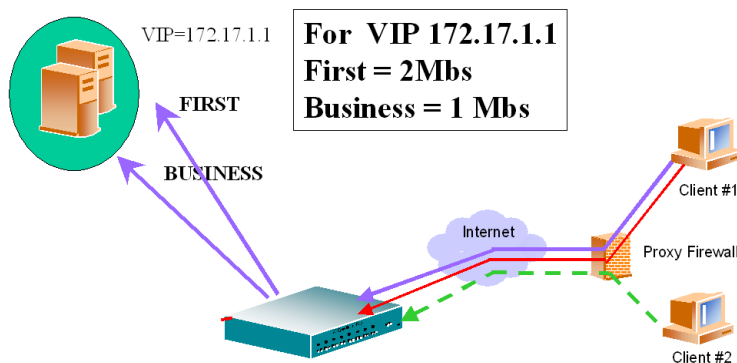


Figure 9-6 Cookie-Based Bandwidth Management

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 WebSwitch 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 watch 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`) 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.

Operational Keys

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.

Configuring Bandwidth Management

The following procedure provides general instructions for configuring BWM on the switch. Specific configuration examples begin on [page 238](#).

1. Configure the switch as you normally would for Server Load Balancing. 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 Chapter 1.

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.

<pre>>> # /cfg/bwm/on</pre>	<i>(Turn BWM on)</i>
-----------------------------------	----------------------

3. Select a bandwidth policy.

Each policy must have a unique number from 1 to 64.

<pre>>> Bandwidth Management# pol 1</pre>	<i>(Select bandwidth policy 1)</i>
---	------------------------------------

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 256K-1000M.

NOTE – For rates less than 1 Mbps, append a “K” suffix to the number.

<pre>>> Policy 1# hard 6</pre>	<i>(Set “never exceed” rate)</i>
<pre>>> Policy 1# soft 5</pre>	<i>(Set desired bandwidth rate)</i>
<pre>>> Policy 1# resv 4</pre>	<i>(Set committed information rate)</i>

5. (Optional) Set the 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 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. Enable the BWM contract.

```
>> BWM Contract 1# ena (Enables this BWM contract)
```

12. Classify the frames for this contract and assign the BWM contract to the filter.

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# ../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.

13. 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.

14. On the switch, save your new configuration changes.

```
>> Bandwidth Management# save (Save for restore after reboot)
```

15. On the switch, check the BWM information.

```
>> Bandwidth Management# /info/bwm <contract number> (View BWM information)
>> Bandwidth Management# /stats/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.

Additional Configuration Examples

Examples are provided for the following:

- User/Application Fairness
- Preferential Services: [page 241](#)
- Security Management: [page 249](#)

User/Application Fairness Example

Bandwidth Management can be applied to prevent heavy bursters from locking out other users, such as in preventing the following:

- Customers using broadband access (such as DSL) from blocking dial-up customers
- Customers from the same hosting facility locking out each other because of flash crowd
- FTP from locking out Telnet
- Rate limit 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 processing 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.

```
>> BWM Contract 1# pol 1 (Assign policy 1 to BWM Contract 1)
```

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                                (Make your changes active)
>> Port 2# /cfg/bwm/cur                          (View current BWM settings)
```

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                    (Save for restore after reboot)
```

14. 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.

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.

Website Preference Example

In the following example, there are two Websites, A.com and B.com. BWM is configured to give preference to traffic sent to Website 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 Chapter 1.

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 (Set "never exceed" rate)
>> Policy 1# soft 8 (Set desired bandwidth rate)
>> Policy 1# resv 5 (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 (Select BWM Contract 1)
>> BWM Contract 1# name a.com (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 (Set "never exceed" rate)  
>> Policy 2# soft 15 (Set desired bandwidth rate)  
>> Policy 2# resv 10 (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 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 Web OS 8.3 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 Web OS Command Reference, Chapter 7, “The Configuration Menu” 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 “alteaon.”**

```
>> # /cfg/slb/url/lb/add alteaon
```

To see the URL path ID assigned for the string “alteaon,” execute the `cur` command:

```
>> Server Loadbalance Resource# cur
Number of entries: 2
1: any, cont 1024
2: alteaon, cont 3
```

2. **Allocate bandwidth for each string.**

To achieve this, assign a BWM contract to a defined string.

```
>> Server Loadbalance 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 (PIP) 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
>> # ../port 2/pip 12.12.12.12
>> # proxy ena
```

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 Website has a single Virtual 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/url/lb/add <URL path ID>
```

Example:

```
>> # /cfg/slb/url/lb/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/url/lb/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
>> # ../port 2/pip 12.12.12.12
>> # proxy ena
```

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 load balancing can be performed.

5. Enable Server Load Balancing.

```
>> # /cfg/slb/on
```

Scenario 2: In this scenario, the Website has multiple Virtual IP addresses and the same user classification or multiple sites use the same string name. In this scenario, there are two VIPs, 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 9-7 on page 248](#).

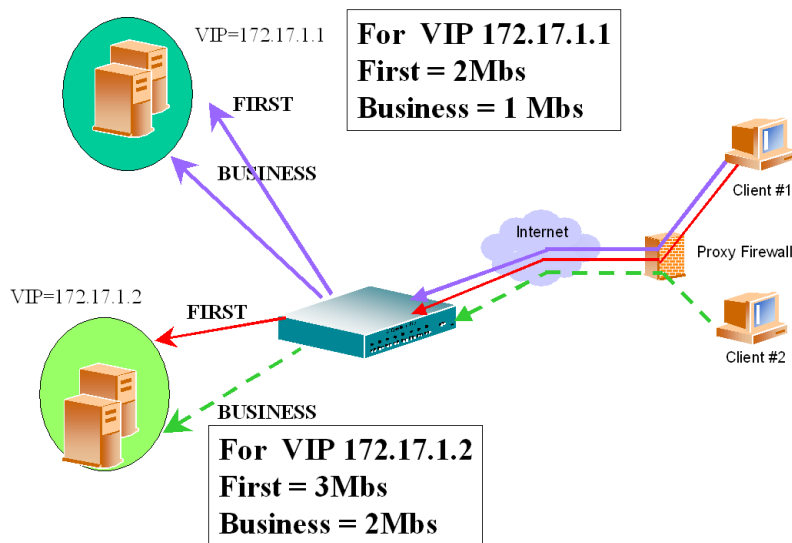


Figure 9-7 Cookie-Based Preferential Services

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/url1/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 flooding of “necessary evil” packets by limiting the rate of TCP SYN, Ping, and other disruptive packets and alerting/logging 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 you normally would for SLB (see [Chapter 1](#)):

- 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 Kbytes.

<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.

<pre>>> Bandwidth Management# /cfg/bwm/cont 1</pre>	<i>(Select BWM contract 1)</i>
<pre>>> BWM Contract 1# name icmp</pre>	<i>(Select contract name “icmp”)</i>

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.

CHAPTER 10

Health Checking

Content-intelligent WebSwitches allow Webmasters to customize server health checks to verify content accessibility in large Web sites. As the amount of content grows and information is distributed across different server farms, flexible, customizable content health checks are critical to ensuring end-to-end availability.

Alteon WebSwitches, running server load balancing, monitor the servers in the real server group and the load-balanced application(s) running on them. If a session 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 WebSwitch can remove the individual service from the load-balancing algorithm without affecting other services provided by that server.

Alteon WebSwitches monitor the health of servers and applications using the health-checking mechanisms described in this chapter.

Health-Check Parameters for Real Servers

By default, the 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. Both the health-check interval and the number of retries can be modified.

NOTE – Health checks, when used in conjunction with a virtual server configured with multiple services/groups, are done sequentially. As a result, the actual health-check interval could vary significantly from the value set for it using the `inter` parameter.

<pre>>> # /cfg/slb/real real-server-number</pre>	<i>(Select the real server)</i>
<pre>>> Real server# inter 4</pre>	<i>(Check real server every 4 seconds)</i>
<pre>>> Real server# retry 6</pre>	<i>(If 6 consecutive health checks fail, declare real server down)</i>

Health-Check Types

Layer 4 health-check support includes TCP and application-specific health checks and ASCII script-based health checks. Layer 3 health checking is available for ICMP. Each health-check type is summarized below:

TCP Health Checks

TCP health checks are useful to verify 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 packet.

ICMP Health Checks

The Layer 3 echo - echo reply health check is used for UDP services or when ICMP health checks are configured.

Wireless Session Protocol Content Health Checks

When devices communicate with the WAP gateway using the unencrypted WSP, they can either establish a connection-oriented WSP session, or work in a connectionless WSP mode. This feature provides a connectionless WSP health check for WAP gateways. Future releases will cover WTLS health checks.

Overview

Connectionless WSP runs on UDP/IP protocol, port 9200. Therefore, Alteon WebSystems' switches can be used to load balance the gateways in this mode of operation.

Web OS 8.3 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.

Enabling WSP Health Checks

1. Enable the WSP health check type from the `/cfg/slb/group` menu.

```
>> # /cfg/slb/group 1/health wsp
```

Configuring WAP 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
Current WSP port: 9200
New pending WSP port: 9200
```

5. Set the offset value.

```
>> WAP Health Check# offset 0
Current WSP receive offset:      0
New pending WSP receive offset: 0
```

6. Because WAP gateways are UDP-based and operate on a UDP port, configure a 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              (Set the health check type)
Current health check type:      tcp
Enter health check type:      wsp
```

8. Apply and save the configuration.

```
>> Virtual Server 1 9200 Service# apply
-----
Apply complete; don't forget to "save" updated configuration.
>> Virtual Server 1 9200 Service# save
Request will first copy the FLASH "active" config to "backup",
then overlay FLASH "active" with new config.
Confirm saving to FLASH [y/n]: y
New config successfully saved to FLASH.
```

Hostname for HTTP Content 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	/cfg/slb/virt/service	9 characters
Domain name	dname	/cfg/slb/virt	35 characters
Server group health-check field	content	/cfg/slb/group	34 characters

If the `HOST:` header is required, an HTTP/1.1 GET will occur, otherwise an HTTP/1.0 GET will occur.

Example 1:

```
hname    = compute
dname    = alteonwebsystems.com
content  = index.html
```

Health check is performed using:

```
GET /index.html HTTP/1.1
Host: compute.alteonwebsystems.com
```

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    = compute
content  = index.html
```

Health check is performed using:

```
GET /index.html HTTP/1.1
Host: compute
```

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  = //compute/index.html
```

Health check is performed using:

```
GET /index.html HTTP/1.1
Host: compute
```

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. 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*).

<code>>> # /cfg/slb/group real-server-group-number</code>	<i>(Select the real server group.)</i>
<code>>> # slb/group/health imap</code>	<i>(Specify the type of health checking to be performed.)</i>
<code>>> # slb/group/content username:password</code>	<i>(Specify the IMAP username:password value.)</i>

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.

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 `NAS_IP` parameter in the RADIUS content health check
- ❑ RADIUS health check using real server port configured, that is, the `rport`
- ❑ Variable-length RADIUS secret password. Supports less than 16 octets and up to 32 octets

RADIUS is stateless and uses UDP as its transport protocol. To support RADIUS health checking, the network administrator must configure two parameters in the switch: the `/cfg/slb/secret` value and 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.)
```

- 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.

Script-Based Health Checks

Using this feature, 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 websites

The expect string can be any string in the entire response. Web OS 8.3 supports the following capacity for a single switch:

- # bytes per script = 1024
- # scripts per switch = 8
- # health-check statements (HTTP GET and expect strings) = approximately 10 to 15

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 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 will be doing HTTP 1.1 pipelining, you'll need to individually open and close each response in the script.

- Each script should start with the command “open port xxx” where xxx is the 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 websites 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 Web server you are done 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, type “rem” to remove the last typed script line entered. If you need to remove more than one line, type “rem” for the 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 and 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 RFC, as applicable. This helps ensure that if the customer changes server software, 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, such as tasks where the interval for load balancing is exceeded, or the health check will fail.

Script Configuration Examples

Example 1: 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."

Example 2: GSLB URL Health Check

In earlier Web OS releases, each remote GSLB site's VIP 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 VIP_1 and the following RIPs:

- RIP_1 and RIP_2: "images"
- RIP_3 and RIP_4: "html"
- RIP_5 and RIP_6: "cgi" and "bin"
- RIP_7: which is VIP_2, "any"

Site #2 with VIP_2 and the following RIPs:

- RIP_1 and RIP_2: "images"
- RIP_3 and RIP_4: "html"
- RIP_5 and RIP_6: "cgi" and "bin"
- RIP_7: which is VIP_1, "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 RIP_1 and RIP_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.

You should configure the remote RIP (the VIP of the remote site) to accept “any” URL requests. The purpose of the first GET is to check if RIP_1 or RIP_2 is up; that is, to check if the remote site has at least one server for “images” content. Either RIP_1 or RIP_2 will respond to the first GET health check.

If all the RIPs are down, RIP_7 (the VIP 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 respond 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
```

HTTPS/SSL Health Check

The `sslh` health-check option on the Real Server Group Menu (`/cfg/slb/group/health/sslh`) 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 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 various 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.

Failure Types

Service Failure

If a certain number of connection requests for a particular service fail, the session 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 but, 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 the “Service Failed” state, the session 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 but, 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 11

Secure Switch Management

To limit access to the switch's Management Processor 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 Web OS Browser-Based Interface (BBI). This will also help prevent spoofing or attacks on the switch's TCP/IP stack.

The allowable management IP address range is configured using the system `mnet` and `mmask` options available on the command line interface System Menu (`/cfg/sys`).

NOTE – The `mnet` and `mmask` commands in the `/cfg/slb/adv` menu are used for a different purpose.

When an IP packet reaches the Management Processor, the source IP address is checked against the range of addresses defined by `mnet` and `mmask`. If the source 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 silently.

Example: Assume that the `mnet` is set to 192.192.192.0 and the `mmask` is set to 255.255.255.128. This defines the following range of IP addresses: 192.192.192.0 to 192.192.192.127.

- 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 Management Processor.
- 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).

NOTE – When the `mnet` and `mmask` Management Processor filter is applied, RIP updates received by the switch will be discarded if the source IP address of the RIP packet(s) falls outside the specified range. This can be corrected by configuring static routes.

Secure Switch Management

Secured 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 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 and SSL. Web OS supports the encryption of management information, using SSH, on AD4 and A184 Web-Switches.

Authentication and Authorization

NOTE – While authentication and authorization (AA) protocols and servers are designed to authenticate remote dial-up users (in addition to authorizing remote access capabilities to users), this overview is focussed on using the AA model to authenticate and authorize remote administrators for managing a switch.

The AA model 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 remote access server, not the back-end server and database.

Two prominent “AA” protocols used to control dial-up access into networks are Cisco's TACACS+ (Terminal Access Controller Access Control System) and Livingston Enterprise's RADIUS (Remote Authentication Dial-In User Service). Web OS 8.3 supports only the RADIUS authentication method.

Components

The required components for authorization and authentication are listed below:

- A remote administrator
- The switch with authentication and authorization protocol support, acting as a client in the AA model
- A back-end authentication and authorization server that performs the following functions:
 - Authenticates remote administrators
 - Checks the remote administrator's authorization to access the switch
 - Optionally, tracks and logs the administrator's activity while logging on
- An AA database that contains information about authorized administrators and their specific capabilities and privileges

Authorization and Authentication Procedure

The steps below describe the process that a remote administrator would go through to get authenticated and authorized for managing a switch, using the RADIUS AA protocol.

- 1. Remote administrator connects to the switch and provides his/her user name and password.**
- 2. Using the RADIUS protocol, the switch sends requests for authentication and authorization to the authentication/authorization server.**
- 3. The authentication/authorization server checks the user name/password combination against its user ID database.**
- 4. Using the RADIUS protocol, the authentication/authorization server instructs the switch to grant or deny the administrator access to the switch, based on its defined capabilities and privileges.**

RADIUS Authentication

RADIUS is an access server authentication, authorization, and accounting protocol used to secure remote access to networks and network services against unauthorized access.

RADIUS is comprised of three components:

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

The operation of RADIUS authentication and authorization protocol is similar to the “AA” model described above. The switch, acting as the RADIUS client, will communicate to the RADIUS server to authenticate and authorize a remote administrator using the protocol definitions specified in RFC 2138. Transactions between the client and RADIUS server are authenticated through the use of a shared secret, which is never 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.

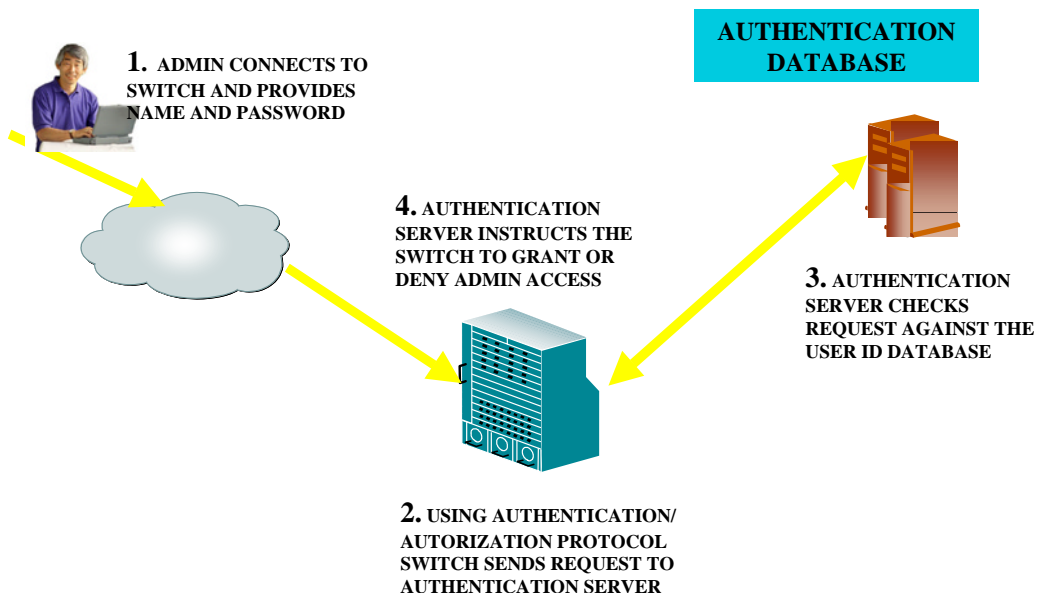


Figure 11-1 Authentication and Authorization: How It Works

RADIUS Authentication Features in Web OS

- Support of RADIUS client on the switch, based on the protocol definitions in RFC 2138.
- An option for the administrator to enable/disable support of RADIUS authentication and authorization.
- The default is to disable the use of RADIUS for authentication and authorization.
- The RADIUS secret password can be up to 32 bytes. It can be less than 16 octets.
- Support of a “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.

- RADIUS server retry and timeout values are user-configurable. The parameters are:
 - Time-out value = 1-10 seconds
 - Retries = 1-3

The switch will time-out if it does not see response from the RADIUS server in (1-3) seconds. The switch will also automatically retry to the RADIUS servers before it declares the server is down.
- Support of user-configurable RADIUS application port. The default is 1645/udp based on RFC 2138.
- Network administrator can define privileges for one or more specific users to access the switch at the RADIUS user database. The following user accounts listed in [Table 11-1](#) can be defined in the RADIUS server dictionary file:

Table 11-1 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 Web 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 Web 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

When the user logs in, the switch authenticates his/her level of access by sending the RADIUS Access-Request, 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. The file name of the dictionary is RADIUS vendor-dependent. The following user privileges are Alteon WebSystems proprietary definitions.

Table 11-2 Alteon WebSystems User Access Levels

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

- SecurID support, provided RADIUS server can do ACE/Server client proxy. The password is the PIN number, plus the tokencode of securID card.

Secure Shell (SSH) and Secure Copy (SCP)

Although a remote network administrator can manage the configuration of an Alteon WebSwitch via Telnet, this method does not provide a secure connection. Using Secure Shell (SSH) and Secure Copy (SCP), messages between a remote administrator and the switch use secure tunnels so that the data on the network is encrypted and secured.

NOTE – SSH/SCP features are supported only on the AD4 and A184 WebSwitches and can only be configured via the console port, using the command line interface. When SSH is enabled, SCP is also enabled.

SSH (Secure Shell) is a protocol that enables a remote administrator to securely log into another computer over a network to execute management commands. All the data sent over the network using SSH is encrypted and secured. Using SSH gives administrators an alternate way to manage the switch, one that provides strong security.

SCP (Secure Copy) is typically used to securely copy files from one machine to another. SCP uses SSH for encryption of data on the network. On an Alteon WebSwitch, 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
Administrator identification using NAME/PASSWORD
- Authorization of remote administrators
Determine the permitted actions
Customize service for individual administrators
- Encryption of management messages
Messages between remote administrator and switch are encrypted
- Secure copy support

NOTE – The Web OS implementation of SSH is based on SSH version 1.5 and supports SSH-1.5-1.x.xx. SSH clients of other versions (especially Version 2) will not be supported.

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)
-

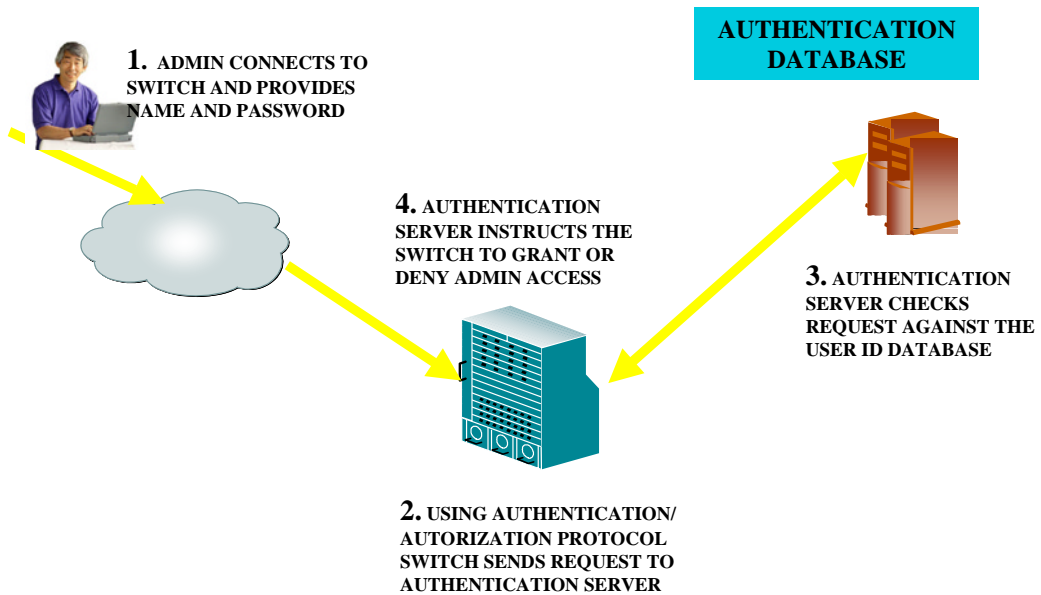


Figure 11-2 Secure Switch Management: How It Works

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

Encryption of Management Messages

The supported encryption and authentication methods for both SSH and SCP are listed below:

Server Host Authentication:	Client RSA authenticates the switch in the beginning of every connection.
Key Exchange:	RSA
Encryption:	3DES-CBC, DES
User Authentication:	local password authentication, RADIUS, SecurID (via RADIUS, for SSH only; that is, not applied to SCP)

SCP Services

NOTE – Administrator privileges are required to perform SCP commands.

Four SCP commands are supported in this service: `getcfg`, `putcfg`, `putcfg_apply`, and `putcfg_apply_save`.

- `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 will be automatically executed at the end of `putcfg` to notify the remote client of the difference between the new and the current configurations
- `putcfg_apply` will run the `apply` command after the `putcfg` is done
- `putcfg_apply_save` will save 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, a 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 switch. The server key is 768 bits and is used to make it impossible to decipher a captured session by breaking into the switch at a later time.

When the SSH server is first enabled and applied, the switch will automatically generate the host and server keys and will then store them into the flash.

NOTE – The WebSwitch 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.

The commands to generate these two keys manually are `/cfg/sys/sshd/hkeygen` and `/cfg/sys/sshd/skeygen`. Again, the host or server key will be automatically stored into flash after generated.

When the switch reboots, it will retrieve the host and server keys from the flash. If these two keys are not available in the flash, and if the SSH server feature is enabled, the switch will automatically generate them during the system reboot.

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

```
>> # /cfg/sys/sshd/interval <n>
```

where n (number of hours) must be in the range (0–24) and a value of 0 denotes that RSA server key auto-generation is disabled. When n is greater than 0, the switch will autogenerate the RSA server key every n hours; however, RSA server key generation will be skipped if the switch is busy doing other key or cipher generation when the timer expires.

Radius Authentication

SSH/SCP is integrated with RADIUS authentication. After the RADIUS server is enabled in 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 log-in, which is not provided by the SSH connection.

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

To log in using SSH without difficulties, you need to use a special username, “ace,” to log in and bypass the SSH authentication. After an SSH connection is established, you will then be prompted to enter the username and password (the SecurID authentication is being performed now). You will need to provide your actual username and the token in your SecurID card as a regular Telnet user would do in order to log in.

To use SCP, you need to use SCP-only administrator’s password (that is, the `scpadm` option under the `/cfg/sys/sshd` menu) to bypass the checking of SecurID. 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 without additional authentication required.

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 since the switch will recognize him as the SCP-only administrator and only allow the administrator access to SCP commands.

Configuring SSH/SCP

NOTE – SSH/SCP parameters can be configured only via the console port, using the Command Line Interface (CLI). The switch SSH daemon uses TCP port 22 only and is not configurable.

To enable/disable the SSH/SCP feature, use this command:

```
>> # /cfg/sys/sshd/on
```

To set the interval of RSA server key auto-generation, use this command:

```
>> # /cfg/sys/sshd/interval <n>
```

where n (number of hours) must be in the range (0–24) and a value of 0 denotes that RSA server key auto-generation is disabled. When n is greater than 0, the switch will auto-generate the RSA server key every n hours; however, RSA server key generation will be skipped if the switch is busy doing other key or cipher generation when the timer expires.

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

```
>> # /cfg/sys/sshd/ena
>> # /cfg/sys/sshd/dis
```

To view the current SSH/SCP-related configuration, use this command:

```
>> # /cfg/sys/sshd/cur
```

To view the difference between the new and current configurations, use this command:

```
>> # diff
```

To apply the pending changes from the new configuration, use this command:

```
>> # apply
```

NOTE – If SSH/SCP is enabled and an **apply** command is issued, the switch will automatically generate the RSA host and server keys if they are not available. It will take several minutes to complete this process.

To save the current configuration to flash, use this command:

```
>> # save
```

Usually, there will be no need to generate manually the RSA host and server keys. However, you may still do so by using the following commands:

```
>> # /cfg/sys/sshd/hkeygen           Generates the host key.
>> # /cfg/sys/sshd/skeygen           Generates the server key.
```

NOTE – These two commands will take effect immediately without the need of an **apply** command being issued.

Some Supported Client Commands

NOTE – Up to four simultaneous Telnet/SSH/SCP connections are supported on a switch.

- To login to the switch:
ssh <switch_ip> or **ssh -l** <username> <switch_ip>
- To download the switch configuration using SCP:
scp <switch_ip>:getcfg <local_filename>
- To upload the configuration to the switch:
scp <local_filename> <switch_ip>:putcfg

Some examples are listed below:

```
>> # ssh 205.178.15.157
>> # ssh -l dleu 205.178.15.157
>> # scp 205.178.15.157:getcfg ad4.cfg
>> # scp ad4.cfg 205.178.15.157:putcfg
```

where *205.178.15.157* is the IP address of the switch.

Please also note that **apply** and **save** commands are still needed after the last command (**scp ad4.cfg 205.178.15.157:putcfg**) is issued. Or, instead, you can use the following commands to avoid the “apply and save” issue:

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



CHAPTER 12

High Availability

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 a redundancy for all vital network components.

Alteon WebSystems switches support high-availability network topologies through an enhanced implementation of the Virtual Router Redundancy Protocol (VRRP).

The Web OS implementation of VRRP supports three modes of high availability: active-standby, active-active, and 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 WebSystems extensions to VRRP. Each is briefly summarized below, with a pointer to where you'll find more information.

■ Active-Standby

In an active-standby configuration, two WebSwitches 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 and behaves as a backup for 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. For a detailed description of this approach, refer to [page 281](#).

■ Active-Active

In an active-active configuration, two WebSwitches provide redundancy for each other, with both active at the same time for the same services. For a detailed description of this approach, refer to [page 282](#).

■ Hot-Standby

VRRP has been extended to support hot-standby failover configurations. 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. For a detailed description of this approach, refer to [page 282](#).

Failover Methods: An Overview

With service availability becoming a major concern on the Internet, service providers are increasingly deploying Internet traffic control devices such as WebSwitches 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 typical hot-standby configuration is shown in the figure below:

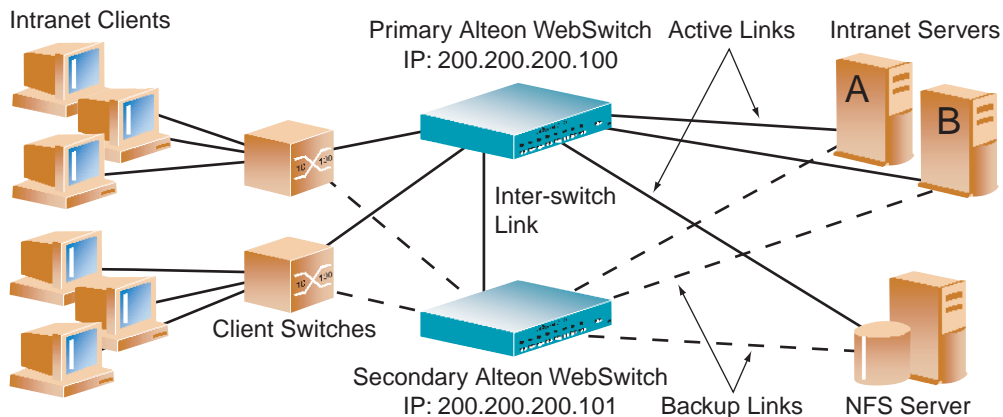


Figure 12-1 A Non-VRRP, Hot-Standby Configuration

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 WebSwitch 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 WebSystems' redundancy configurations are based on the extensions to VRRP that it has developed to support Layer 4 switching services such as server load balancing (SLB), to support active operation of interfaces (at Layer 3) and services (at Layer 4) across multiple switches at the same time, and to interact with Spanning Tree to control frame path.

Alteon WebSystems switches support three approaches to providing high availability and redundancy: *active-standby*, *active-active*, and a new *hot-standby* configuration. Each is described in this section.

Active-Standby Redundancy

In an *active-standby configuration*, shown in Figure 12-2, both switches can support active traffic. However, since services are not shared across the switches. Each switch can be active for some number of services, such as IP routing interfaces or load-balancing VIP addresses, and act as a standby for other services on the other switch.

NOTE – In an active-standby configuration, the same service cannot be active simultaneously on both switches.

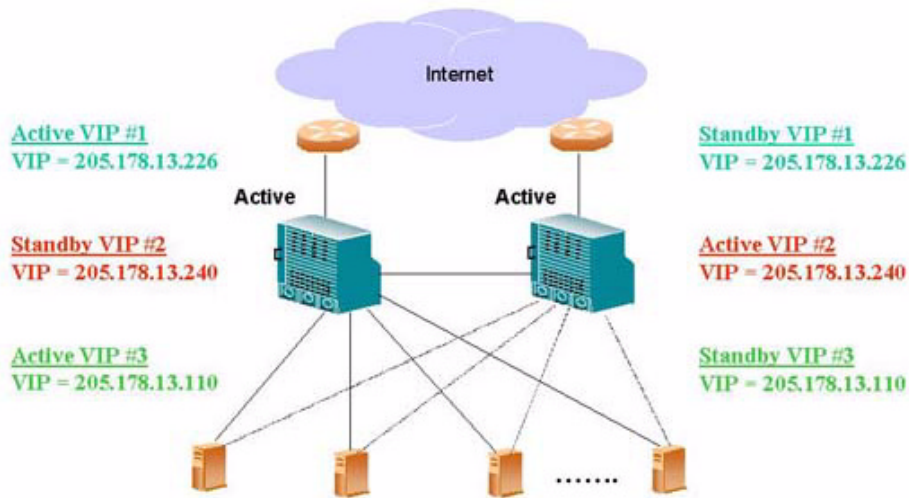


Figure 12-2 Active-Standby Redundancy

Active-Active Redundancy

Alteon WebSystems 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 12-3](#), 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).

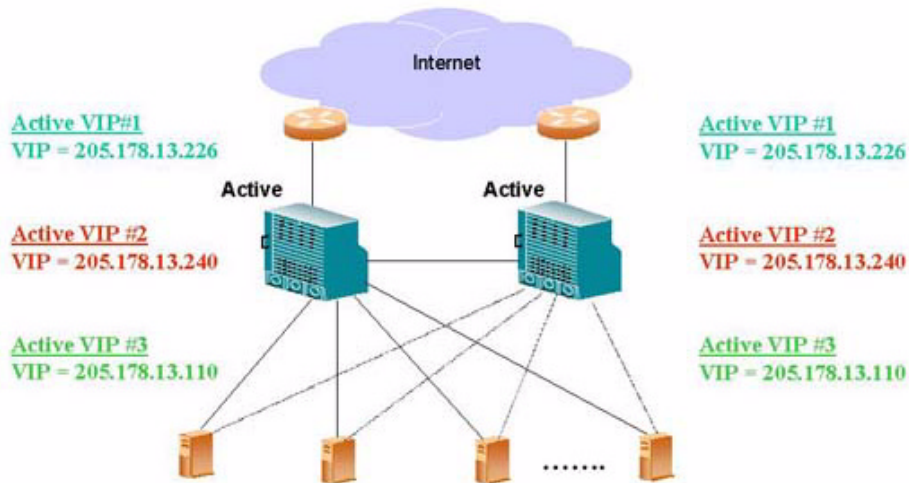


Figure 12-3 Active-Active Redundancy

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

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 12-4 on page 283](#).

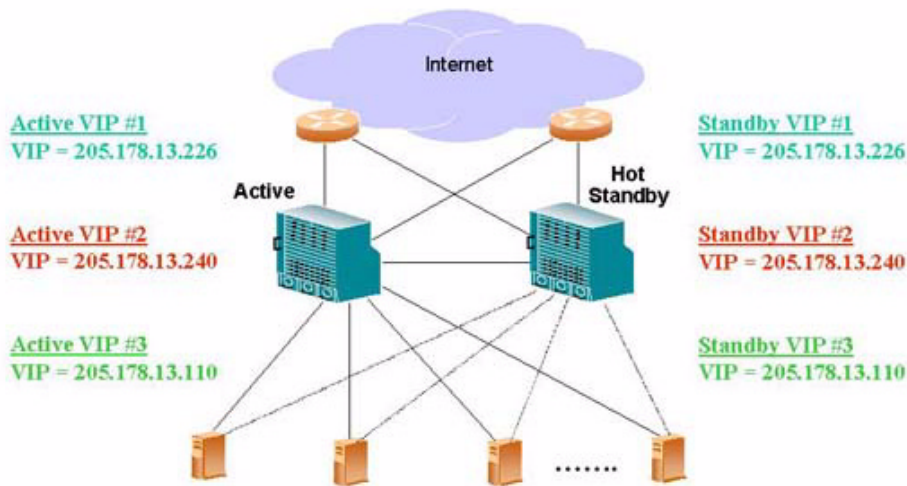


Figure 12-4 Hot-Standby Redundancy

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 (VRs) 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, meaning all virtual routers share the same state, master or backup. They cannot transition from master to backup or vice versa individually, only as a whole.

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 VIP 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. However, the advertisements are also used to help bridges learn the virtual router MAC address. Since all of the virtual routers can have different virtual router identifiers (VRIDs), we must rotate the MAC source address of the advertisement to ensure that the bridges learn all of the virtual router MAC addresses.

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 "hotstan" using `/cfg/slb/port x/hotstan`.
- Links that are used by VRRP to deliver updates are configured as "intersw," or inter-switch link (not to be confused with Cisco's ISL). The command to configure one or more ports as interswitch link is `/cfg/slb/port x/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 VRs tracking value is "2" 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.

Configuration Synchronization

The final piece in configuring a high-availability solution includes the addition of synchronization options to simplify the manual configuration synchronization. Configuration options have been added to refine what is synchronized, to whom, and to disable synchronizing certain configurations. These 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.

VRRP Overview

To give you the background necessary to understand the operation of Alteon WebSystems' redundancy configurations, this section describes VRRP operation and the Alteon-specific extensions to VRRP.

VRRP enables redundant router configurations within a LAN, providing alternate router paths for a host to eliminate single points of failure within a network. This is accomplished by configuring the same virtual router IP address and ID number on each participating VRRP-capable routing device. 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.

Since 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. VRRP-based redundancy significantly reduces administrative overhead when compared to redundancy schemes that require hosts to be configured with multiple default gateways.

VRRP Components

Each physical router running VRRP is known as a *VRRP 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 Extensions to VRRP” on page 12-290](#). 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.

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.

One, but not more than 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, when up, 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*.

Within each virtual router, one of the VRRP routers is selected to be the virtual router master. See [“Determining Which VRRP Router Is the Master”](#) on page 288 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 it’s priority (explained below).

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 above points are illustrated in [Figure 12-5](#). The Alteon WebSystems switches in the diagram have been configured as VRRP routers. They form a *virtual interface router (VIR)*.

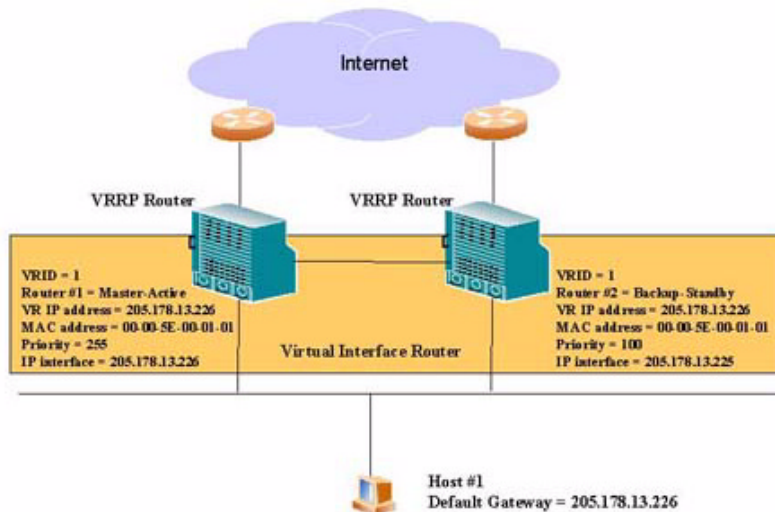


Figure 12-5 VRRP Router Example 1

The switch on the left of [Figure 12-5](#) 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. The switch on the right 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 12-5](#) 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. Since, in this example, the master is also the virtual interface router's IP address owner, 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, since this switch is not the owner, it does not have a real interface configured with the virtual interface router's IP address.

Determining Which VRRP Router Is the Master

Each VRRP router that is not an owner is configured with a priority between 1–254. Per the VRRP standard, an owner has a priority = 255. A bidding process determines which VRRP router is or becomes the master: the VRRP router with the highest priority. Since owners have a priority higher than the range permitted for non-owners, the IP address owner, if any, is always the master for the virtual interface router, as long as it is available.

The master periodically sends out 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. That VRRP router then takes over as 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's more than one) to become the master.

NOTE – If the master is healthy but communication between it and the backup has failed, there will be then two masters within the virtual router. To prevent this from happening, it is strongly recommended that redundant links be used between the switches that form a virtual router.

If, at any time, a backup router determines that it has higher priority than the current master, it can preempt the master, unless it is configured not to do so. In preemption, the backup assumes the role of master and begins to send its own advertisements. The current master will see that the backup has higher priority and will stop functioning as the master.

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. VRRP can also be used to implement active-standby configurations. In the example shown in [Figure 12-6](#), the switch on the left is the master for the virtual interface router with VRID = 1 and backup for the virtual interface router with VRID = 2. The switch on the right 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.

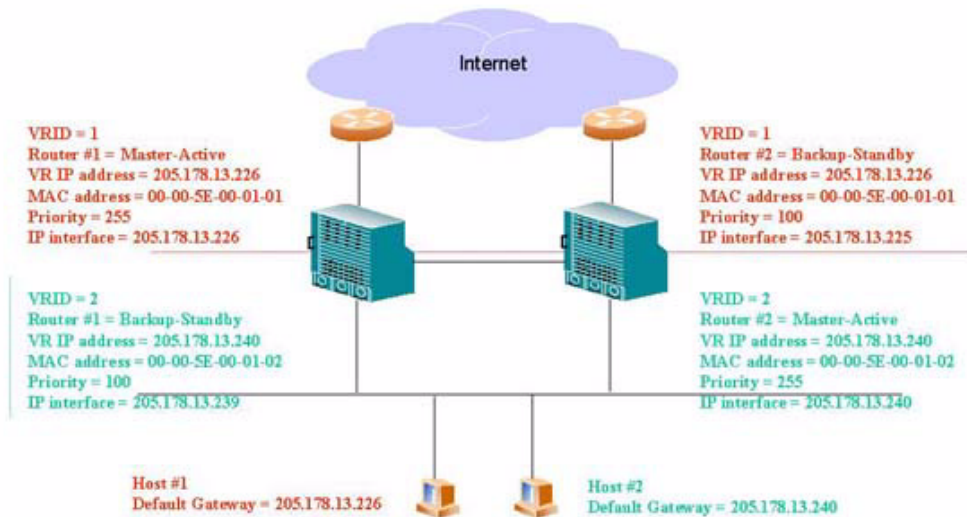


Figure 12-6 VRRP Router Example 2

Alteon Extensions to VRRP

This section describes VRRP enhancements that are implemented in Web OS.

Virtual Server Routers

Web OS supports *virtual server routers*, which extend the benefits of VRRP to VIP addresses used to perform SLB.

Virtual server routers operate for VIP 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 processes packets destined for the VIP address and responds to ARP requests. The master sends periodic advertisements. If a backup does not receive an advertisement within a specified period, it initiates the bidding process to determine which VRRP router 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.

One difference between virtual server routers and virtual interface routers is that the concept of an IP address owner does not apply to virtual server routers. All virtual server routers are *renters*. For a virtual server router, the master always responds to ICMP ping requests if *sharing* (see [page 291](#)) is not enabled. If sharing is enabled, the switch where the ping request initially enters the virtual server router responds.

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/Active-Active Failover

Web OS supports *sharing* of interfaces at both Layer 3 and Layer 4, as shown in [Figure 12-7](#). With sharing, an IP interface or a VIP address can be active simultaneously on multiple switches, enabling active-active operation.

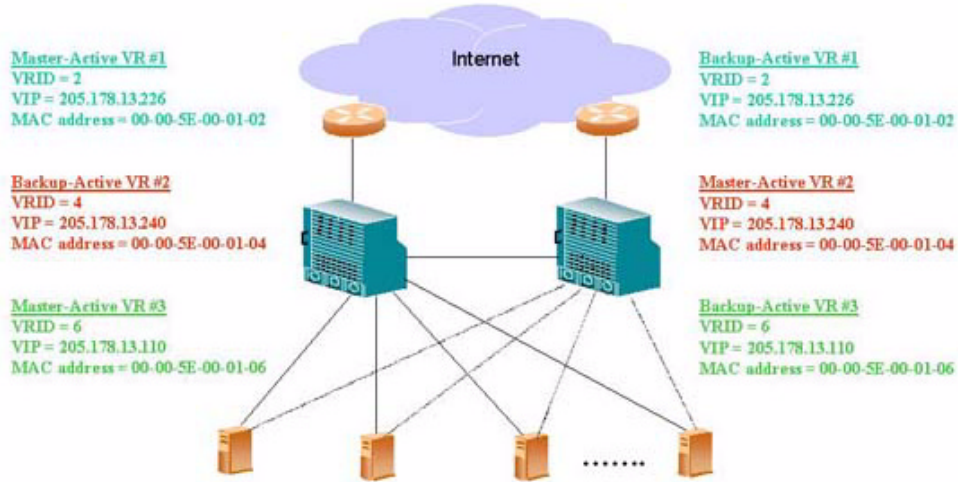


Figure 12-7 Active-Active High Availability

When sharing is used, incoming packets are processed by the switch on which they enter the virtual router. This 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 VIP address, it does determine which switch sends advertisements and responds to ARPs sent to the virtual router's IP address.

Alteon WebSystems 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

Web 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 and to ensure 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.

Tracking only affects hot standby and active-standby configurations. It does not have any effect when sharing; that is, when active-active configurations are used.

Web OS can track the attributes listed in [Table 12-1](#):

Table 12-1 VRRP Tracked Parameters

Parameter	Description
Number of virtual routers in master mode on the switch	Useful for making sure 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	An IP interface is considered active when there is at least one active port on the same VLAN. This helps elect the virtual routers with the most available routes as the master. 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	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.
Number of physical switch ports that have active Layer 4 processing on this switch	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.

Table 12-1 VRRP Tracked Parameters

Parameter	Description
Number of healthy real servers behind the VIP address that is the same as the IP address of the virtual server router on the switch	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	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.

Redundancy Configurations

Alteon WebSystems 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 (below)
- Active-active virtual server router configuration on [page 296](#)
- Active-active server load balancing on [page 298](#)
- Hot-standby configuration on [page 306](#)

Active-Standby Virtual Server Router Configuration

Figure 12-8 shows an example configuration where two Alteon WebSwitches 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 VIP.

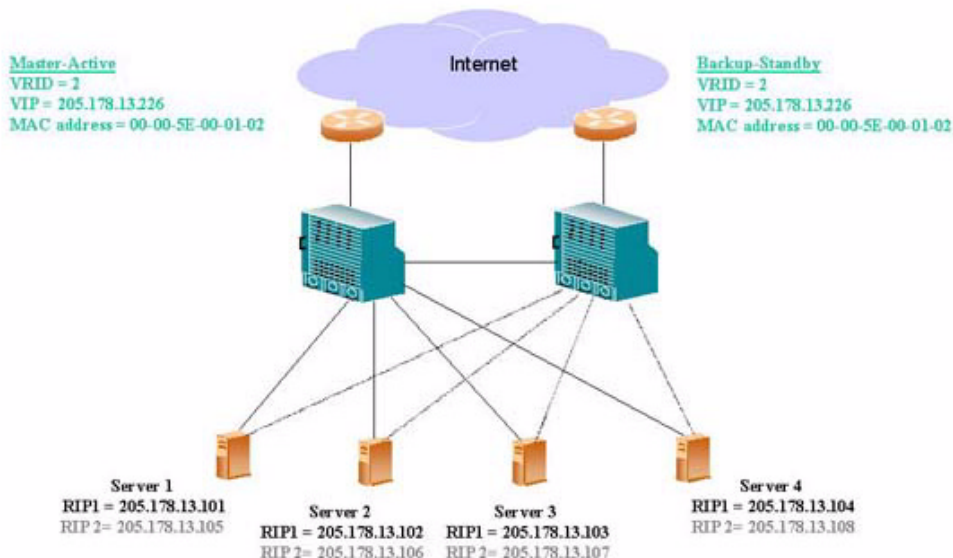


Figure 12-8 Active-Standby High Availability Configuration

Although this example shows only two switches, there is no limit on the number of switches used in a redundant configuration. It's 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 of them should use the VIP address described in Step 4.

- 2. Define all needed filters.**

For Web OS release 5.2.13, this must be done on both switches. For later Web OS releases, the filters may be configured on one switch and pushed to 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 the switch on the left 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 the switch.**

This includes the VRID = 2, the VIP = 205.178.13.226 and the priority. Enable tracking and set the parameters appropriately (refer to [“Configuring Tracking” on page 311](#)). Make sure to disable sharing.

- 5. Synchronize the SLB and VRRP configurations by pushing the configuration from the switch on the left to the one on the right.**

Use the `/oper/slb/sync` command.

- 6. Change the real servers in the right-hand switch’s configuration to RIP = 205.178.13.105, RIP = 205.178.13.106, RIP = 205.178.13.107 and RIP = 205.178.13.108.**

Adjust the right-hand switch’s priority appropriately (see [“Configuring Tracking” on page 311](#)).

In this example, with the left-hand switch as the Master, if a link between the left-hand switch and a server fails, the server will fail health checks and be taken out of the load-balancing algorithm. Assuming that tracking is enabled and is configured to take into account the number of healthy real servers for the Virtual Router’s VIP address, the left-hand switch’s priority will be reduced. If it is reduced to a value lower than the right-hand switch’s priority, the right-hand switch will assume the role of Master.

NOTE – In this case, all active connections serviced by the left-hand switch’s VIP will be severed.

If the link between the left-hand (master) 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. The right-hand (backup) switch will act as a Layer 2/3 switch and forward all traffic destined to the VIP to the left-hand switch.

If the entire left-hand (master) switch fails, the protocol used to distribute traffic between the routers, such as Open Shortest Path First (OSPF), will reroute traffic to the right-hand router. The right-hand (backup) switch/router will detect that the master has failed because it will stop receiving advertisements. The backup will then assume the master's responsibility of responding to ARPs and issuing advertisements.

Active-Active VIR and VSR Configuration

Figure 12-9 shows an example configuration where two Alteon WebSwitches are used as VRRP Routers in an active-active configuration implementing a virtual server router. As noted earlier, this is the preferred redundant configuration.

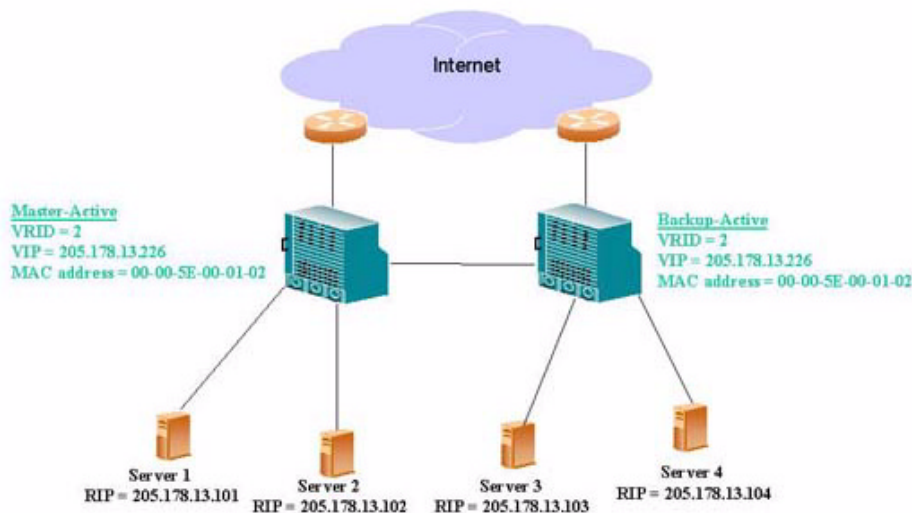


Figure 12-9 Active-Active High-Availability Configuration

Although this example shows only two switches, there is no limit on the number of switches used in a redundant configuration. It's 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, both load balanced packets are sent to the VIP, 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, perform the following switch configuration procedure:

- 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 of them should use the VIP address described in Step 4.

- 2. Define all needed filters.**

For Web OS release 5.2.13, this must be done on both switches. For later Web OS releases, the filters may be configured on one switch and pushed to 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 the switch on the left ([Figure 12-9](#)) 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 and 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.**

This includes 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 the switch on the left to the one on the right.**

Use the `/oper/slb/sync` command.

- 6. Reverse the roles of the real servers and their backups in the right switch’s configuration.**

Make RIP = 205.178.13.103 and RIP= 205.178.13.104 the real servers and RIP = 205.178.13.101 and RIP = 205.178.13.102 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.

This also happens if an entire master switch fails. The backup will detect this fact because it will stop receiving advertisements. In this case, the backup will assume the master’s responsibility of responding to ARPs and issuing advertisements.

Think carefully before setting maxconns in this configuration. Information about maxconns 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 maxconns limits defined for it on both switches.

Active/Active Server Load Balancing Configuration

In this example, you will set up four virtual servers (VIPs), each load balancing two servers providing one service (for example, HTTP) per VIP.

You will be load balancing HTTP, HTTP-S, POP3, SMTP, and FTP. Each protocol will be load balanced via a different VIP. You could load balance all of these services on one VIP, but in this example, four distinct VIPs will be used to illustrate the benefits of active/active failover. You will set up one switch, dump out the configuration script (also called a text dump), edit it, and dump the 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 Web OS Browser-Based Interface (BBI) or the Command Line Interface (CLI) for configuration.

Procedure #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 1.

NOTE – On Alteon WebSystems switches, you are not restricted to configuring only one subnet per VLAN.

To configure the IP interfaces for this example, enter the following commands from the CLI:

>> # /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 this sequence of commands for each interface listed below:

- IF #1 10.10.10.10
- 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, you need to 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).

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

Repeat this command for the second VLAN.

- VLAN #1 - IF #5 - physical ports connected to upstream switches have membership
- VLAN #2 - IFs #1,2,3,4 - physical ports connected to downstream switches have membership

3. Disable Spanning Tree.

Spanning Tree can be turned off in this configuration. Reboot the switch to turn Spanning Tree off, after disabling it using the following command:

```
>> # /cfg/stp off                             (Globally disable STP)
```

4. Enable IP forwarding.

You need to enable IP forwarding if the VIP and RIPv 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). If you are in doubt as to whether or not to enable IP forwarding, enable it. In our example, the VIP and RIPv are indeed on different subnets, so enable this feature using the following command:

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

Procedure #2: SLB Configuration

1. Define the Real Servers

The RIPs 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:

>> # /cfg/slb/real 1	<i>(Server A is real server 1)</i>
>> Real server 1 # rip 10.10.10.5/24	<i>(Assign Server A IP address)</i>
>> Real server 1 # ena	<i>(Enable real server 1)</i>

Repeat this sequence of commands for the following real servers:

- RIP #1 10.10.10.5/24
- 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	<i>(Select real server group 1)</i>
>> Real server group 1# add 1	<i>(Add real server 1 to group 1)</i>
>> Real server group 1# add 2	<i>(Add real server 2 to group 1)</i>

Repeat this sequence of commands for the following real server groups:

- Group #1 – Add RIP #1 and #2
- 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 VIPs 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.

```
>> 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
>> 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 #1 200.200.200.100 will load balance HTTP (Port 80) to Group 1
- VIP #2 200.200.200.101 will load balance HTTP-S (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'ing) of the RIP back to the VIP. 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 1# client ena                  (Enable client 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)
```

Procedure #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 VIPs. This is what tells the switch that these are service on the Virtual server 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 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 #1 - Priority 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) to track on is Layer 4 ports (l4pts).

Use the following command:

```
>> Virtual server 4# /cfg/vrrp/vr 1/track l4pts
```

This command sets priority tracking parameter for virtual router 1, electing the virtual router with most available ports as the master router. Repeat this command for the following virtual routers:

- VR #1 - Track l4pts VR #5 - Track l4pts
- 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.

Configuring Switch 2

Option 1:

This is the preferred option.

1. Create an IP interface and enable it.

```
>> # /cfg/ip/if 1/addr 10.10.10.11/ena
```

2. Create a sync peer and enable it.

```
>> # /cfg/slb/sync/peer 1/addr 10.10.10.10/ena
```

3. Apply and save.

4. Check /info/ip to see whether the interface are up. You should be able to ping switch 1. Ping 10.10.10.10.

5. From switch 1, issue the command /oper/slb/sync to synchronize the switch 1 configuration to switch 2.

6. When the config sync is successful, login to switch 2 and issue the command

```
>> # /cfg/vrrp/vvl/vrid 1/if 1/prior 101
```

7. Apply and save.

Option 2:

Use the following procedure to dump the configuration script (text dump) out of Switch 1:

If you have been using the Web OS Browser-Based Interface (BBI) to configure the switch, you need a serial cable that is a DB-9 Male to DB-9 Female, straight-through (not a null modem) cable. Connect to the switch from your computer with the cable. 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.

If you are using HyperTerminal, only the Baud Rate and Flow Control options need to be changed from the default settings. Once you connect to the switch, start logging your session in HyperTerminal (transfer/capture text). 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. Stop logging your session (transfer/capture text/stop).

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 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.

Go to the second switch. Any configuration on it needs to be deleted by resetting it to factory settings, using the following command:

```
>> # /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`).

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.

NOTE – To use hot-standby redundancy, peer switches must have an equal number of ports.

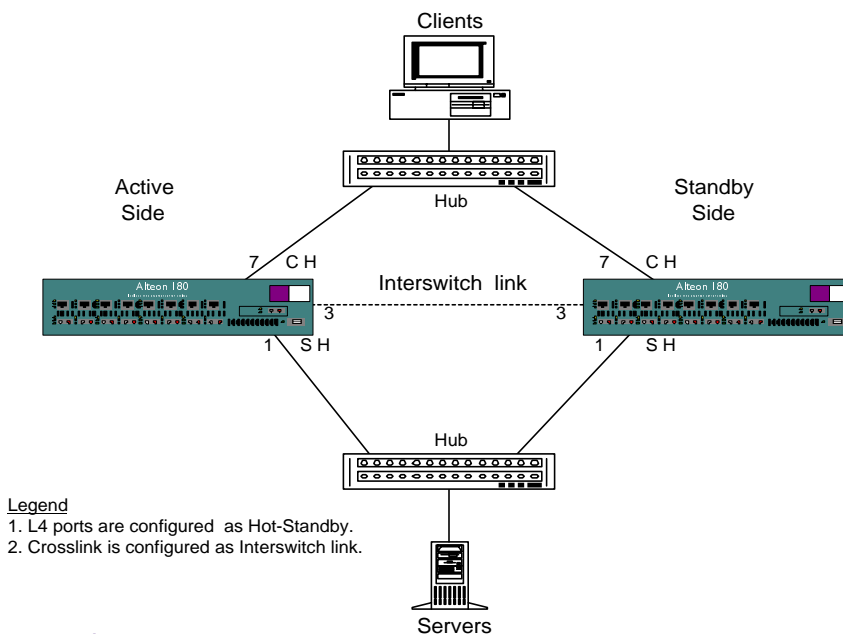


Figure 12-10 Hot-Standby Configuration

Figure 12-10 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 just VRRP.

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 to 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 VLAN's and enabling IP forwarding to route between them.

By reducing complexity to a single subnet and not requiring routing (L3), 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 12-10](#)). STP doesn't 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.**

```
/cfg/slb/port 1
server ena
hotstan ena
/cfg/slb/port 2
hotstan ena
/cfg/slb/port 3
intersw ena
/cfg/slb/port 7
client ena
hotstan ena
```

2. **From the VRRP menu, enable VRRP group mode; then enable hot-standby.**

```
/cfg/vrrp/on
/cfg/vrrp/hotstan enabled
/cfg/vrrp/group ena
```

3. **Sync the VRRP, SLB and filter settings to the other switch (same ports).**

```
/oper/slb/sync
```

NOTE – Switches peering with each other must have an equal number of ports.

4. **Turn off STP after verifying the network is stable.**

```
/cfg/stp off
save
/boot/reset
```

NOTE – You must reboot the for the hot-standby configuration to take effect.

Virtual Router Deployment Considerations

To prevent network problems when deploying virtual routers, you should review the issues described in this section.

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.

VRRP Active/Active Synchronization

The old 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:

```
/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.

NOTE – In Web OS 8.3, the `sync` command also copies IP proxy settings to the target switch creating duplicate IP addresses on your network. To correct this problem, you must reconfigure each IP proxy on the target switch to use a unique IP address.

Using the /oper/slb/sync Command

For user convenience, it is possible to push 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.

Using Web OS 8.3, all server load balancing and VRRP parameters can be pushed using the `/oper/slb/sync` command. You must configure and enable the peer switch using the command `/cfg/slb/sync/peer 1/ip address/ena` (of the other VRRP switch interface.)

Peers must be configured and enabled on both the switches. Both these switches should have the same admin password. Port specific configurations such as Proxy IPs, filters, and Layer 4 port states (for example, client and server), VRRP priorities, and Bandwidth Management configurations can be prevented from synchronizing by using the menus in the command: `/cfg/slb/sync`.

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. As a result, 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 (and unpleasant) results may occur if the `/oper/slb/sync` command attempts to configure a non-existent port or applies an inappropriate configuration to a port.

Using the /cfg/slb/sync Command

To synchronize the configuration between two switches, a peer must be configured on each switch. Switches being synchronized must use the same administrator password. Peers are sent SLB, FILT, and VRRP configuration updates using `/oper/slb/sync`.

VRRP, STP, and Failover Response Time

VRRP active/active failover is significantly different from the hot-standby failover method supported in previous releases. As shown in [Figure 12-11](#), active-active configurations can introduce loops into complex LAN topologies.

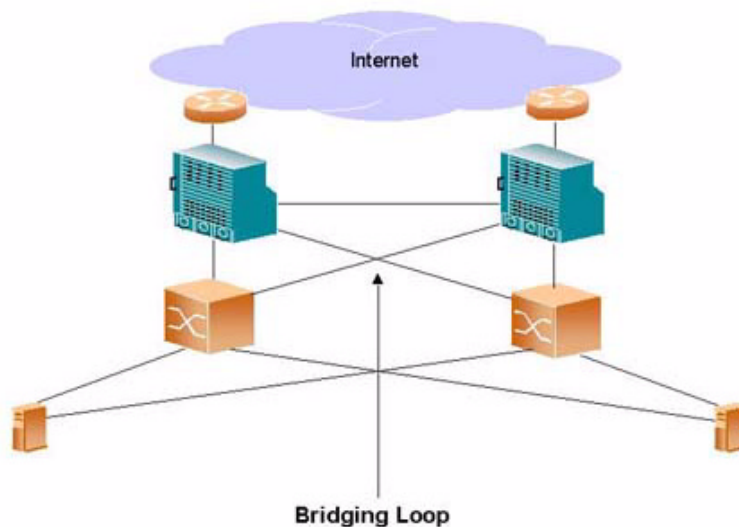


Figure 12-11 Loops in Active-Active Configuration

Using STP to Eliminate Loops

VRRP generally requires STP to be enabled in order to resolve bridge loops that usually occur in cross-redundant topologies, as shown in [Figure 12-12](#). In this example, a number of loops are wired into the topology. STP resolves loops by blocking ports where looping is detected.

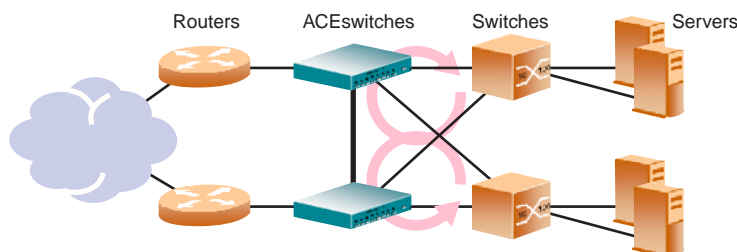


Figure 12-12 Cross-Redundancy Creates Loops, But STP Resolves Them

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 12-13](#):

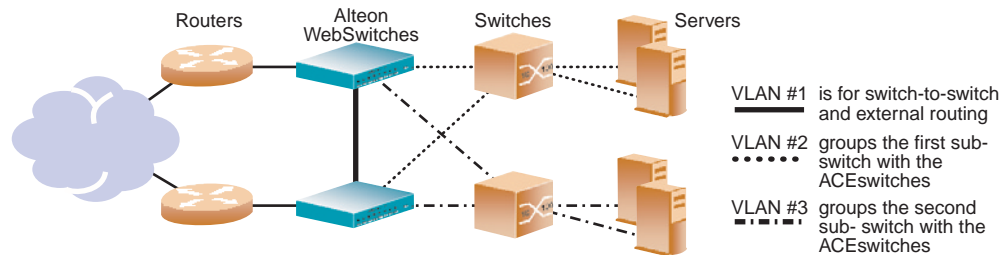


Figure 12-13 VLANs can be used to Create Non-Looping Topologies

This topology allows STP to be disabled. On the Alteon WebSwitches, IP routing allows traffic to cross VLAN boundaries. The servers use the Alteon WebSwitches 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).

VRRP Virtual Router ID Numbering

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 Tracking

Proper tracking configuration largely depends on user preferences and network environment. Consider the configuration shown in [Figure 12-8 on page 294](#). Assume that the user wants the following behavior in their network:

- The switch on the left will be the master router upon initialization.
- If the switch on the left is the master and it has one active server fewer than the switch on the right, it remains the Master.
- The user wants this behavior because s/he believes running one server down is less disruptive than bringing a new master online and severing all active connections in the process.
- If the switch on the left is the master and it has two or more active servers fewer than the switch on the right, the switch on the right becomes the Master.
- If the switch on the right is the Master, it remains the master even if servers are restored to the point on the left-hand switch where the left-hand switch has one fewer or an equal number of servers.
- If the switch on the right is the master and it has one active server fewer than the switch on the left, the switch on the left becomes the Master.

The user can implement this behavior by configuring tracking as follows:

- 1. Set the priority for the left-hand switch to the default value of 100.**
- 2. Set the priority for the right-hand switch 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 that is the same as the IP address of the virtual server router on the switch and set the value = 6.**

Initially, the switch on the left will have a priority of 100 (base value) + 5 (since it will initially be the Master) + 24 (4 active real servers x 6 per real server) = 129. The switch on the right will have a priority of 96 (base value) + 24 (4 active real servers X 6 per real server) = 120.

If one server attached to the left-hand switch fails, the left-hand switch's priority will be reduced by 6 to 123. Since 123 is greater than 120 (the right-hand switch's priority), the left-hand switch will remain the Master.

If a second server attached to the left-hand switch fails, the left-hand switch's priority will be reduced by 6 more to 117. Since 117 is less than 120 (the right-hand switch's priority), the right-hand switch will become the Master. At this point, the left-hand switch's priority will fall by 5 more and the right-hand switch's will rise by 5 because the switches are tracking how many Masters they are running. So, the left-hand switch's priority will settle out at 112 and the right-hand switch's priority at 125.

When both servers are restored to the left-hand switch, that switch's priority will rise by 12 (2 healthy real servers X 6 per healthy server) to 124. Since 124 is less than 125, the right-hand switch will remain the Master.

If, at this point, a server fails on the right-hand switch, its priority will fall by 6 to 119. Since 119 is less than 124, the left-hand switch will become the Master. Its priority will settle out at 129 (since it's now the Master) while the right-hand switch's will drop by 5 more to 114.

We see from the above 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.

Stateful Failover of L4 and L7 Persistent Sessions

Overview

Web OS 8.3 also provides stateful failover of content-intelligent persistent session state and Layer 7 persistent session state. This includes SSL session state, HTTP cookie state, and Layer 4 persistent and FTP session state. Providing stateful failover enables network administrators to mirror their Layer 7 and Layer 4 persistent transactional state on the peer switch.

NOTE – Stateful failover for Layer 7 persistency (for example: SSL session ID persistence-based server load balancing, URL and Cookie-based server load balancing) requires Direct Access Mode (DAM) to be enabled. Stateful failover for Layer 4 (for example: FTP-based server load balancing) does not require DAM to be enabled.

To provide stateful failover, the state of the connection and session table must be shared between the switches in high-availability configurations. With Virtual Matrix Architecture (VMA) enabled, all URL and cookie-parsing information is stored in the session table on port 9. Sharing this information between switches is necessary to ensure the persistent session goes back to the same server.

NOTE – Stateful failover is only supported in active-standby mode with VMA enabled.

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 12-14](#). 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 (switch 2) becomes active.
2. The incoming request is redirected to switch 2.
3. When the user clicks Submit again, the request is forwarded to the correct server.

Even though switch 1 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 switch 2.

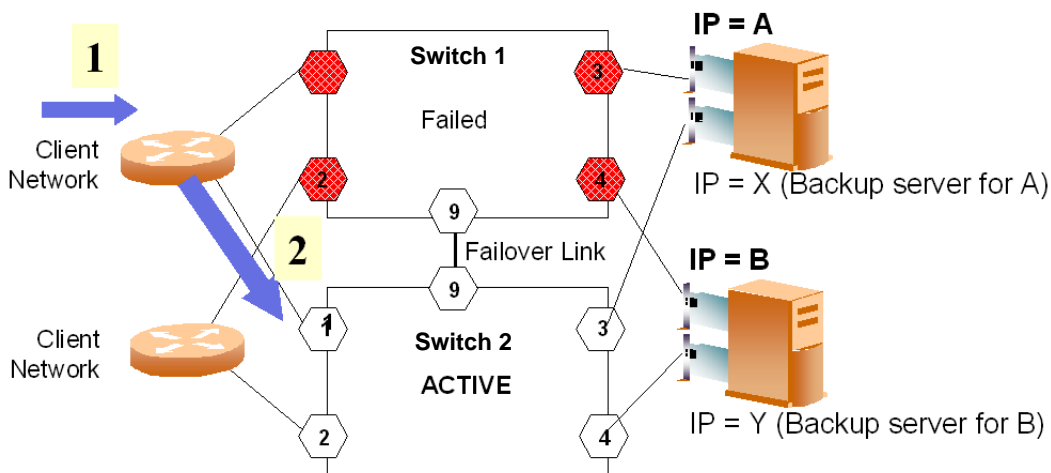


Figure 12-14 Stateful Failover Example when the Master Switch Fails

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)
```

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.

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.

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
```



CHAPTER 13

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.

Basic VLANs can be configured during initial switch configuration. See the *Web OS Command Reference* for more comprehensive VLAN information.

VLAN ID Numbers

Web OS supports up to 246 VLANs per switch. Even though the maximum number of VLANs supported at any given time is 246, each can be identified with any number between 1–4094.

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 below).

Each port in the switch has a configurable default VLAN number, known as its *PVID*. The factory default value of 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 1–4094.

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

VLAN Tagging

Web 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 Spanning-Tree

When Spanning Tree Protocol (STP) is enabled on the switch, it detects and eliminates logical loops in a bridged or switched network. 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.

If you configure the switch with Spanning-Tree, there will be a single instance of Spanning Tree per switch, regardless of the number of configured VLANs in an enabled state.

VLANs and the IP Interfaces

Careful consideration must be made when creating VLANs within the switch, such that communication with the switch Management Processor (MP) remains possible where it is required.

Access to the switch for remote configuration, trap messages, and other management functions can only be accomplished from stations on VLANs that include an IP interface to the switch (see “IP Interface Menu” in Chapter 7 of the *Web OS Command Reference*). Likewise, access to management functions can be cut off to any VLAN by excluding IP interfaces from its 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 all have access to switch management features.

VLAN Topologies and Design Issues

By default, the Web OS 8.3 software has a single VLAN configured on every port. This groups all ports into the same broadcast domain. This VLAN has an 802.1Q VLAN PVID of 1. Since in this default only a single VLAN is configured per port, VLAN tagging is turned off.

Since VLANs are most commonly used to create individual broadcast domains and/or separate IP subnets, it is useful for host systems to be able to have presence on more than one VLAN simultaneously. Alteon WebSystems' WebSwitches and ACEnic adapters have the unique capability of being able to support multiple VLANs on a per port or per interface basis, allowing very flexible configurations.

You can configure multiple VLANs on a single ACEnic adapter, with each VLAN being configured through a logical interface and logical IP address on the host system. Each VLAN configured on the 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 multi-VLAN system, you can logically connect users and segments to a host with a single ACEnic adapter that supports many logical segments or subnets.

Example 1: Multiple VLANs with Tagging Adapters

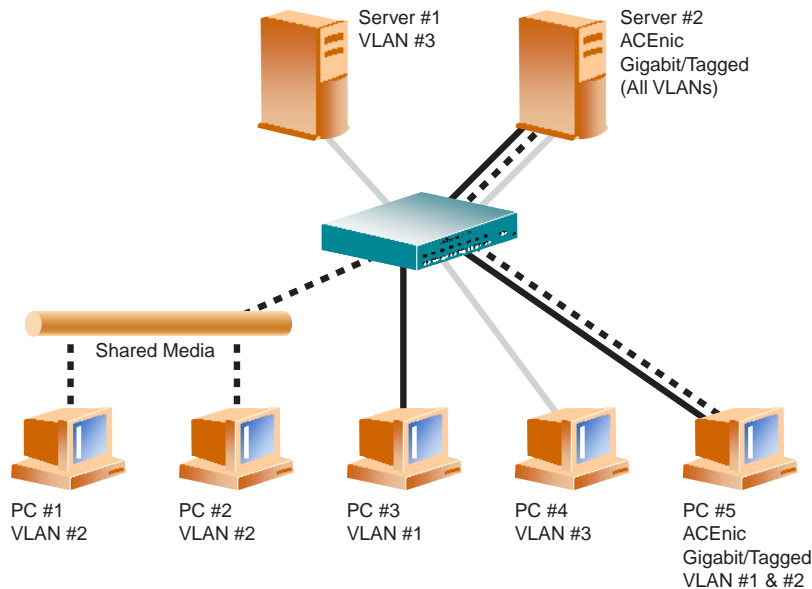


Figure 13-1 Example #1: Multiple VLANs with Tagging ACEnic Adapters

The features of this VLAN are described below:

Component	Description
Alteon Web-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 the VLAN #3 and only has presence in one IP subnet. The port that it is attached to is configured only for VLAN #3, so VLAN tagging is off.
Server #2	A high-use server that needs to be accessed from all VLANs and IP subnets. This server has an Alteon WebSystems ACEnic adapter installed with VLAN tagging turned on. The adapter is attached to one of the Alteon WebSwitch's Gigabit Ethernet ports, that is configured for VLANs #1, #2, and #3 and also has tagging 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 but continues to maintain broadcast separation between all three VLANs and subnets.
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 only communicate with Server #2 and PC #5.
PC #4	A member of VLAN #3, this PC can only communicate with Server #1 and Server #2.
PC #5	A member of both VLAN #1 and VLAN #2, this PC has an Alteon WebSystems' ACEnic Gigabit Ethernet Adapter installed. It is able to communicate with 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 turned on.

NOTE – VLAN tagging is only required on ports that are connected to other Alteon WebSystems switches or on ports that connect to tag-capable end-stations, such as servers with Alteon WebSystems ACEnic Gigabit Ethernet Adapters.

Example 2: Parallel Links with VLANs

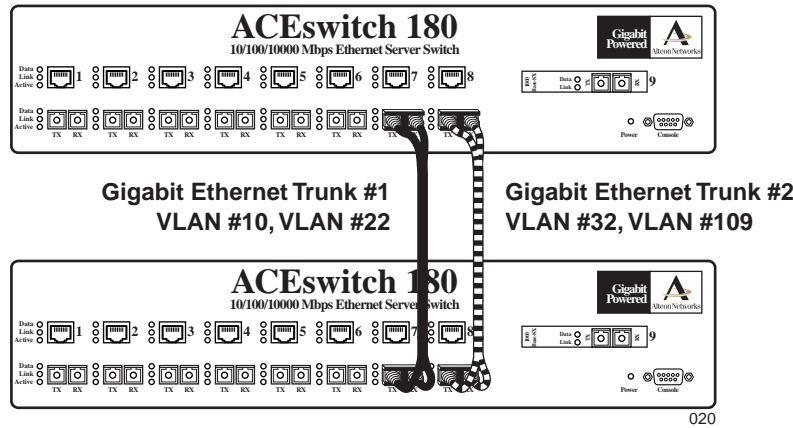


Figure 13-2 Example 2: Parallel Links with VLANs

The following items describe the features of this example:

- Example 2 shows how, through the use of VLANs, it is possible to create configurations where there are multiple links between two switches, without creating broadcast loops.
- Two Alteon WebSystems switches are connected with two different Gigabit Ethernet links. Without VLANs, this configuration would create a broadcast loop, but the STP topology resolution process resolves parallel loop-creating links.
- With VLANs, neither switch-to-switch link shares the same VLAN and, thus, are separated into their own broadcast domains.
- Ports 1 and 2 on both switches are on VLAN 10; ports 3 and 4 on both switches are on VLAN #22. Ports 5 and 6 on both switches are on VLAN #32; and port 9 on both switches is on VLAN #109.
- It is necessary to turn off Spanning Tree on at least one of the switch-to-switch links, or alternately turned off in both switches. Spanning Tree executes on a per-network level, not a per-VLAN level. STP Bridge PDUs will be transmitted out both connected Gigabit Ethernet ports and be interpreted by the connected switch that there is a loop to resolve.
- Spanning Tree is not VLAN-aware. Therefore, any VLAN configuration that might involve a parallel link from an STP perspective must be taken into account during network design. Alteon WebSystems recommends that you avoid topologies such as these, if at all possible.



CHAPTER 14

Jumbo Frames

To reduce host frame processing overhead, the Alteon WebSystems ACEnic adapters and WebSwitches, both running operating Web OS version 2.0 or later, can receive and transmit frames that are far larger than the maximum normal Ethernet frame. By sending one jumbo frame instead of myriad smaller frames, the same task is accomplished with less processing.

The switches and the ACEnic adapter support Jumbo Frame sizes up to 9018 octets. These can be transmitted and received between ACEnic adapter-enabled hosts through the switch across any VLAN.

Isolating Jumbo Frame Traffic using VLANs

Jumbo frame traffic must not be used on a VLAN where there is any device that cannot process frame sizes larger than Ethernet maximum frame size.

Additional VLANs can be configured on the adapters and switches to support non-jumbo frame VLANs for servers and workstations that do not support extended frame sizes. End-stations with an ACEnic adapters installed and attached to switches can communicate across both the jumbo frame VLANs and regular frame VLANs at the same time.

In the example illustrated in [Figure 14-1 on page 324](#), the two servers can handle Jumbo Frames but the two clients cannot; therefore jumbo frames should only be enabled and used on the VLAN represented by the solid lines but not for the VLAN with the dashed lines. Jumbo frames are not supported on ports configured for half-duplex mode.

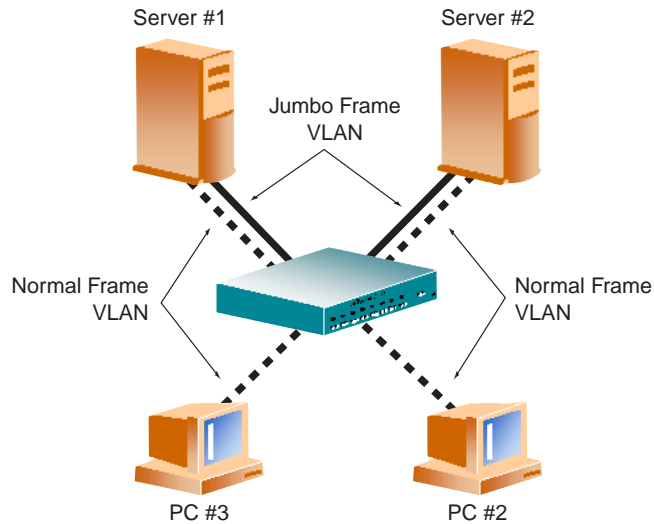


Figure 14-1 Jumbo Frame VLANs

Routing Jumbo Frames to Non-Jumbo Frame VLANs

When IP Routing is used to route traffic between VLANs, the switch will fragment jumbo UDP datagrams when routing from a jumbo frame VLAN to a non-jumbo frame VLAN. The resulting jumbo frame to regular frame conversion makes implementation even easier.



CHAPTER 15

IP Routing

This chapter provides configuration background and examples for using the switch to perform routing functions.

IP Routing Benefits

IP Routing allows the network administrator to seamlessly connect server IP subnets to the rest of the backbone network, using a combination of configurable IP switch interfaces and IP routing options.

The IP Routing feature enhances Alteon WebSystems' server switching solution in the following ways:

- It provides the ability to perform Server Load Balancing (using both Layer 3 and Layer 4 switching in combination) to server subnets which are separate from backbone subnets.
- By automatically fragmenting UDP jumbo frames when routing to non-jumbo frame VLANs or subnets, it provides another means to invisibly introduce jumbo frames technology into the server-switched network.
- It provides the ability to seamlessly route IP traffic between multiple VLANs configured in the switch.

Example of 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. ACElerate powered switches, in fact, are now smart enough and fast enough to perform routing functions on 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:

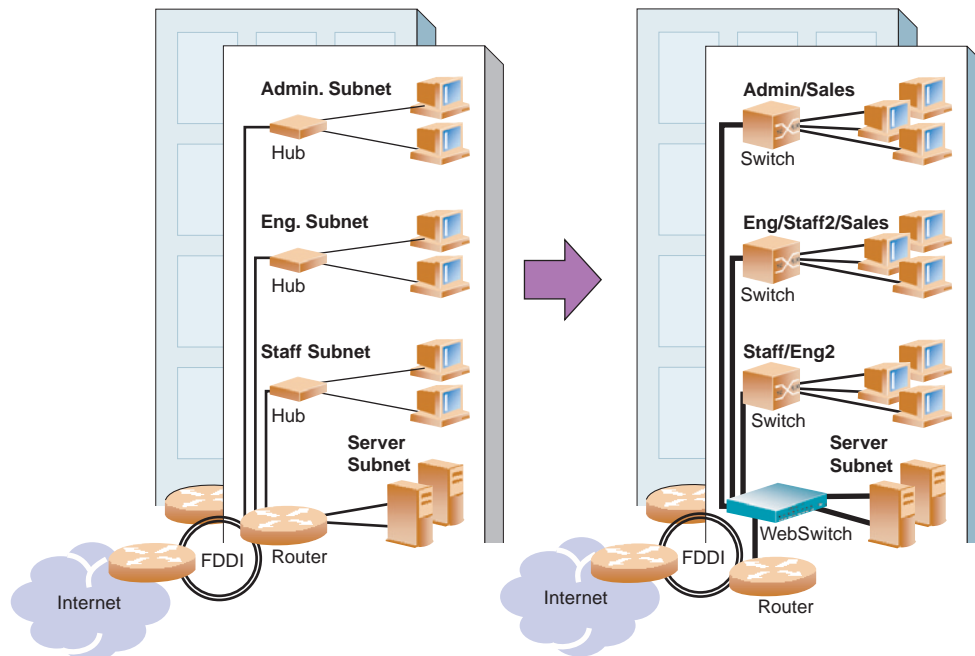


Figure 15-1 The Router Legacy Network

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 hodge-podge 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, worsening any congestion.

Even if every end-station on the network 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 WebSystem switches with built-in IP Routing capabilities. Cross-subnet LAN traffic can now be routed within the Web OS-powered 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 WebSwitch in the example configuration:

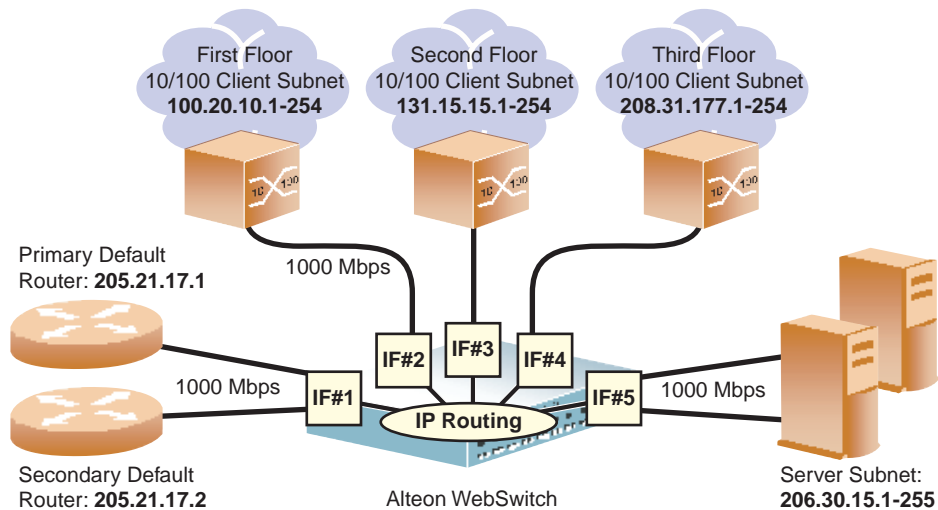


Figure 15-2 Switch-Based Routing Topology

The Alteon WebSwitch 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 relays the packet to the proper destination subnet using Layer 2 switching.

With Layer 3 IP Routing in place on the Alteon WebSystems 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.

As an added benefit, UDP jumbo frame traffic is automatically fragmented to regular Ethernet frame sizes when routing to non-jumbo frame subnets. For instance, this allows servers to communicate with each other using jumbo frames, and to non-jumbo frame devices using regular frames, all transparently to the user.

Example Alteon WebSwitch Configuration for Subnet Routing

Prior to configuration, you must be connected to the switch command line interface as the administrator (see Chapter 2 in the *Web OS Command Reference*).

NOTE – For details about any of the menu commands described in this example, see “IP Configuration” in Chapter 7 of the *Web OS Command Reference*.

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

In our example topology in [Figure 15-2 on page 327](#), the following IP addresses are used:

Table 15-1 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	100.20.10.1-254
#3	Second Floor Client Workstations	131.15.15.1-254
#4	Third Floor Client Workstations	208.31.177.1-254
#5	Common Servers	206.30.15.1-254

2. **On the switch, assign an IP interface for each subnet attached to the switch.**

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

Table 15-2 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	100.20.10.16
IF #3	Second Floor Client Workstations	131.15.15.1
IF #4	Third Floor Client Workstations	208.31.177.2
IF #5	Common Servers	206.30.15.200

These 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 100.20.10.16 (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 208.31.177.2 (Assign IP address for the interface)
>> IP Interface 4# ena (Enable IP interface 4)
>> IP Interface 4# ../if 5 (Select IP interface 5)
>> IP Interface 5# addr 206.30.15.200 (Assign IP address for the interface)
>> IP Interface 5# ena (Enable IP interface 5)
```

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

This 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 (Point to 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 (Point to secondary router)
>> Default gateway 2# ena (Enable secondary default gateway)
```

5. On the switch, enable, apply, and verify the configuration.

```
>> Default gateway 2# ../fwrd (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 any appropriate changes.

6. On the switch, save your new configuration changes.

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

Another Option: Adding VLANs to the Routing Example

The routers, servers, and clients in the example above are all in the same broadcast domain. If limiting broadcasts is desired in your network, you could use VLANs to create distinct broadcast domains. For example, you could create one VLAN for the routers, one for the servers, and one for the client trunks.

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

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

The following table adds ports and VLANs information:

Table 15-3 Subnet Routing Example: Optional VLAN Ports

VLAN	Devices	IP Interface	Switch Port
#1	First Floor Client Workstations	3	1
	Second Floor Client Workstations	4	2
	Third Floor Client Workstations	5	3
#2	Primary Default Router	1	4
	Secondary Default Router	2	5
#3	Common Servers #1	6	6
	Common Servers #2	7	7

2. On the switch, set the default VLAN for each port.

>> # /cfg/port 1	(Select port for First Floor)
>> Port 1# pvid 1	(Set default to VLAN 1)
>> Port 1# ../port 2	(Select port for Second Floor)
>> Port 2# pvid 1	(Set default to VLAN 1)
>> Port 2# ../port 3	(Select port for Third Floor)
>> Port 3# pvid 1	(Set default to VLAN 1)
>> Port 3# ../port 4	(Select port for default router 1)
>> Port 4# pvid 2	(Set default to VLAN 2)
>> Port 4# ../port 5	(Select port for default router 2)
>> Port 5# pvid 2	(Set default to VLAN 2)
>> Port 5# ../port 6	(Select port for common server 1)
>> Port 6# pvid 3	(Set default to VLAN 3)
>> Port 6# ../port 7	(Select port for common server 2)
>> Port 7# pvid 3	(Set default to VLAN 3)

3. On the switch, enable the VLANs.

>> Port 7# /cfg/vlan 1	<i>(Select VLAN 1, the client VLAN)</i>
>> VLAN 1# ena	<i>(enable VLAN 1)</i>
>> VLAN 1# ../vlan 2	<i>(Select VLAN 2, the def. router VLAN)</i>
>> VLAN 2# ena	<i>(enable VLAN 2)</i>
>> VLAN 2# ../vlan 3	<i>(Select VLAN 3, the server VLAN)</i>
>> VLAN 3# ena	<i>(enable VLAN 3)</i>

4. On the switch, 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 15-3 on page 330](#), 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 third floor)</i>
>> IP Interface 4# vlan 1	<i>(Set to VLAN 1)</i>
>> IP Interface 4# ../if 5	<i>(Select IP interface 5 for servers)</i>
>> IP Interface 5# vlan 3	<i>(Set to VLAN 3)</i>

5. On the switch, 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.

6. On the switch, save your new configuration changes.

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

Defining IP Address Ranges for the Local Route Cache

The 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 which will be cached on the switch. The `local network address` is used to define the base IP address in the range which will be cached, and the `local network mask` is the mask which 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 15-4 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 - 255.255.255.255	128.0.0.0	128.0.0.0
205.32.0.0 - 205.32.255.255	205.32.0.0	255.255.0.0

NOTE – All addresses that fall outside the defined range are forwarded to the default gateway. The default gateways must be within range.

Border Gateway Protocol (BGP)

Border Gateway Protocol (BGP) is an Internet protocol that enables routers on a network to share routing information with each other and advertise information about the segments of the IP address space they can access within their network 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 WebSystems switches can advertise their IP interfaces and VIP addresses using BGP, as well as take BGP feeds from up to four BGP router peers. This gives you more resilience and flexibility in balancing traffic from the Internet.

Internal Routing vs. 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 routing protocols such as RIP, RIPv2, and OSPF.

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

External BGP (eBGP) is used to exchange routes between different autonomous systems, while internal BGP is used to exchange routes within the same autonomous system. Internal BGP (iBGP) is one of the internal routing protocols that you can use to do active routing inside your network.

Typically, an AS will have one or multiple “border routers,” peer routers that exchange routes with other ASs, as well as an internal routing scheme enabling every router in that AS to get to 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.

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 had 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 out a path 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’re using, or you will not have connectivity to the host in question.

BGP Failover Configuration

Use the following example to create redundant default gateways for an Alteon switch at an 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 [Table 15-3](#), one Alteon switch is connected to two default gateways, “ISP1” and “ISP2.” The customer negotiates with two ISPs to allow the Alteon switch to use their peer routers as default gateways. The ISP peer routers will then need to announce themselves as default gateways to the switch.

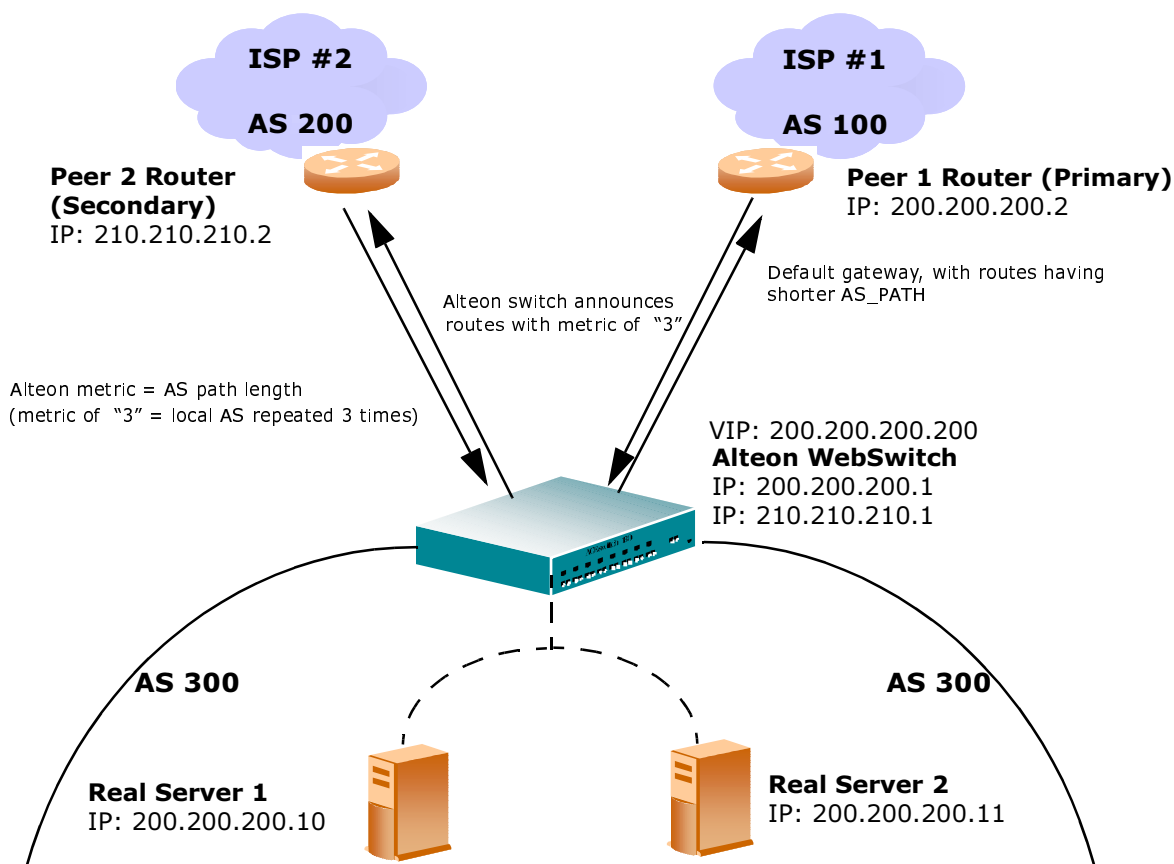


Figure 15-3 BGP Configuration Example

On the 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. ISP2, 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).

This includes the following tasks:

- 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 1.

2. Define the VLANs.

For simplicity, both default gateways are configured in the same VLAN in this example. They 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)
>> vlan 1# ena	(Enable VLAN 1)

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# vlan 1	(Configure VLAN # for this interface)
>> IP Interface 1# /cfg/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)
>> IP Interface 2# vlan 1	(Configure VLAN # for this interface)

4. Enable IP forwarding.

IP forwarding is used for VLAN-to-VLAN (non-BGP) routing. You need to enable IP forwarding 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 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.

```
>> /cfg/ip/bgp/peer 1
>> BGP Peer 1# ena
>> BGP Peer 1# addr 200.200.200.2
>> BGP Peer 1# if 200.200.200.1
>> BGP Peer 1# las 300
>> BGP Peer 1# ras 100
>> BGP Peer 1# /cfg/ip/bgp/peer 2
>> BGP Peer 2# ena
>> BGP Peer 2# addr 200.200.200.2
>> BGP Peer 2# if 210.210.210.1
>> BGP Peer 2# las 300
>> BGP Peer 2# ras 200
>> BGP Peer 2# metric 3
```

(Select BGP peer router 1)
(Enable this peer configuration)
(Set IP address for peer router 1)
(Set IP interface for peer router 1)
(Set local AS. number)
(Set remote AS number)
(Select BGP peer router 2)
(Enable this peer configuration)
(Set IP address for peer router 2)
(Set IP interface for peer router 2)
(Set local AS number)
(Set remote AS number)
(Set AS path length to 3 router hops)

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

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

```
>> BGP Peer 2# apply
>> save
```

(Make your changes active)
(Save for restore after reboot)

BGP-Based Global Server Load Balancing

BGP-based Global Server Load Balancing (GSLB) utilizes the Internet's routing protocols to localize content delivery to the most efficient and consistent site. It does this through the use of a shared IP block that co-exists in each ISP's network and which is then, using BGP, advertised 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 the requests down to that site. In effect, the Internet is making the decision of the “best” location for you, avoiding the need for advanced GSLB.

DHCP Relay

Dynamic Host Configuration Protocol (DHCP) allows hosts (DHCP clients) on an IP network to obtain their configurations from a DHCP server. This reduces the work necessary to administer an IP network. The most significant configuration option the client receives from the server is its IP address (other parameters includes the “generic” filename to be booted, the address of the default gateway, and so on).

Alteon WebSystems' DHCP relay agent eliminates the need to have DHCP/BOOTP servers on every subnet, enabling the network administrator to reduce the number of DHCP servers deployed on the network and 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 Overview

DHCP is described in RFC 2131 and the DHCP relay agent supported on Alteon WebSwitches 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 is an extension of the Bootstrap Protocol (BOOTP), defining mechanisms 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 WebSwitch 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 client, replacing the client source IP (SIP) and destination IP (DIP) address. 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 DIP 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 WebSwitch 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 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.

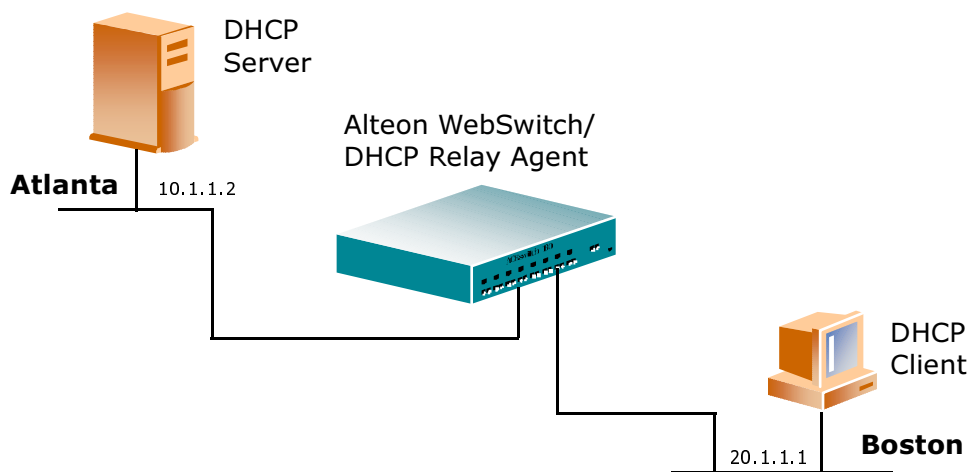


Figure 15-4 DHCP Relay Agent Configuration

In Alteon WebSwitches' implementation, there is no use of primary or secondary servers. The client request is simply forwarded to the BOOTP servers configured on the switch. The use of two servers provides 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 BOOTP relay configuration)
```

Additionally, DHCP Relay functionality can be assigned on per IF interface.

Use the following command to enable the Relay functionality

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


CHAPTER 16

Port Trunking

This chapter provides configuration background and examples for trunking multiple ports together.

Port Trunking Overview

Basics

Trunk groups can provide super-bandwidth, multi-link connections between Alteon WebSystems 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.

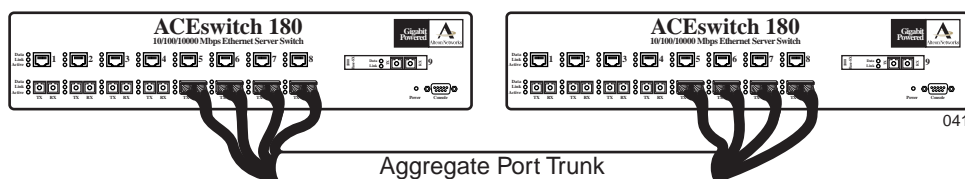


Figure 16-1 Port Trunk Group

When using port trunk groups between two Alteon WebSwitch, for example, the network administrator can create a virtual link between the switches operating up to six Gigabits per second, depending on how many physical ports are combined. The switch supports up to four trunk groups per switch, each with two to six links.

Trunk groups are also useful for connecting an Alteon WebSystems 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. Alteon WebSystems' 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 Web 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 this example, three ports will be trunked between two Alteon WebSwitches.

Prior to configuring each switch in this example, you must connect to the appropriate switch's command line interface as the administrator (see Chapter 2 of the *Web OS Command Reference*).

NOTE – For details about any of the menu commands described in this example, see “Trunk Configuration” in Chapter 7 of the *Web OS Command Reference*.

1. Connect the switch ports that will be involved in the trunk group.

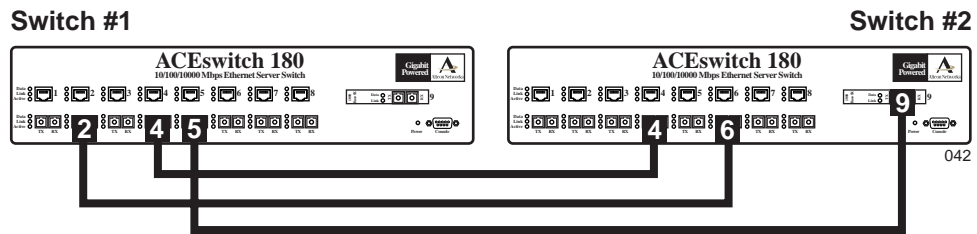


Figure 16-2 Port Trunk Group Configuration Example

2. On switch 1, 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 4                            (Add port 4 to trunk group 1)
>> Trunk group 1# add 5                            (Add port 5 to trunk group 1)
>> Trunk group 1# ena                              (Enable trunk group 1)
```

3. On switch 1, apply and verify the configuration.

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

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

4. On switch 1, save your new configuration changes.

```
>> Trunk group 1# save                            (Save for restore after reboot)
```

5. On switch 2, repeat the process.

>> # /cfg/trunk 3	(Select trunk group 3)
>> Trunk group 3# add 4	(Add port 4 to trunk group 3)
>> Trunk group 3# add 6	(Add port 6 to trunk group 3)
>> Trunk group 3# add 9	(Add port 9 to trunk group 3)
>> Trunk group 3# ena	(Enable trunk group 3)
>> Trunk group 3# apply	(Make your changes active)
>> Trunk group 3# cur	(View current trunking configuration)
>> Trunk group 3# save	(Save for restore after reboot)

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

NOTE – In this example, two Alteon WebSystems 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.

6. 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 four 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 WebSystems' devices must comply with Cisco® EtherChannel® technology.



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 SIP or DIP to another address, network address translation can be said to have taken place. In general, half NAT is when the DIP or SIP is changed from one address to another and full NAT is when both addresses are changed from one address to another. VIP-based load balancing uses half NAT by design since it NAT's the DIP (destination IP address) from the VIP (virtual IP address) to that of one of the RIP's (real servers).
OSPF	Open Shortest Path First: A routing protocol developed for IP networks based on the shortest path first of link-state algorithm.
Preemption	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	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. It 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	Group of real servers that are associated with a VIP or filter.

Redirection or Filter-Based Load Balancing

A type of load balancing; one that operates differently from VIP-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 VIP 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 (either the SIP or DIP). When doing redirection-based load balancing, the DIP is not NAT'ed to that of one of the real servers. Therefore, redirection-based load balancing is designed to be used to load balance devices that normally operate transparently in your network, such as a firewall, spam filter, or transparent Web cache.

RIP (Real Server IP) Address

A real server IP address that the switch load balances to when requests are made to a virtual IP address (VIP).

Server Load Balancing

Basic 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 (RIP) addresses, updates the relevant checksums, and forwards the frame to the server for which it's 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 its MAC address, forcing the packet up to Layer 3. The server would then drop the frame, since the packet would have the DIP of the virtual server (VIP) and not that of the real server (RIP).

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

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 parameters:

- Vrs: Virtual routers in master mode (increments priority by 2 for each)
- Ifs: Active IP interfaces on the 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 IP) Address

An IP address that the switch owns and uses to load balance particular service requests (like HTTP) to other servers.

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 switches must be in a VLAN. If there is more than one VLAN defined on the switch, then the VRRP broadcasts will only be sent out on the VLAN for which the associated IP interface is a member.

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 to ensure devices have a next hop or default gateway that is always available. For example, two or more devices sharing an IP interface are either advertising or listening for advertisements. These advertisements are sent via a broadcast message to address 224.0.0.18.

With VRRP, one switch is considered the master and the other is 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.

VRID (Virtual Router Identifier)

A value between 1 and 255 that is used by each virtual router to create its MAC address and identify its peer with 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 with whom to share.

VIR (Virtual Interface Router)

A VRRP address that is an IP interface address shared between two or more virtual routers.



Index

Symbols

[] 18

Numerics

80 (port) 151
802.1Q VLAN tagging 318, 319

A

ACEnic adapters
 jumbo frames 323
 supporting VLANs 319
active cookie mode (cookie rewrite mode) 207
active-active redundancy 282
 configuration 296
 Server Load Balancing 298
 synchronization 308
active-standby redundancy 281
 configuration 294
administrator account 270
allow (filtering) 51, 53
application health checking 251
application ports 52
application redirection 77
 client IP address authentication 86
 example with NAT 81
 games and real-time applications 86
 non-cacheable sites 86
 non-HTTP redirects for GSLB 161
 proxies 78, 81 to 84
 rport 81, 85
 topologies 79
 web-cache redirection example 77 to 86
authoritative name servers 147
autonomous systems (AS) 333

B

backup servers 37
bandwidth management 230 to 231
 burst limit 234
 classification policies 228
 configuration, general 235 to 237
 configuration, preferential service 241
 configuration, security 249
 configuration, user fairness 238
 cookie-based 232
 data pacing 227
 HTTP header-based 232
 operational keys 234
 precedence 229
 VMA 224
bandwidth policies 225
bandwidth policy rates 226
bandwidth statistics and history 233
Border Gateway Protocol (BGP) 333
 with GSLB 337
Bridge Protocol Data Unit (BPDU) 321
broadcast domains 317, 319, 321, 330

C

CGI-bin scripts 24, 36
Cisco EtherChannel 344
client traffic processing 25
 SLB web balancing example 31
command conventions 18
contacting Alteon WebSystems 19
cookie assignment servers 208 to 211
 configuration examples 208
Cookie-based persistence 204 to 217
 configuring 212
Cookies in HTTP 205
customer support 19

D

data pacing	227
datagram	125
default gateway	327
configuration example	151, 329
default password	270
deny (filtering)	51, 53
dip (destination IP address) for filtering	69, 87
direct real server access	42
Distributed Site State Protocol (DSSP)	146, 151
dmask (destination mask) for filtering	69, 87
domain name	155
domain name server	148
Domain Name System (DNS)	
filtering	56, 59
Global SLB (diagram)	147
round robin	22
dport (filtering option)	57, 82
DSSP. <i>See</i> Distributed Site State Protocol.	
duplex mode for jumbo frames	323
dynamic NAT	66

E

egresses	125
encrypt	125
EtherChannel	341
as used with port trunking	344
external routing	333

F

failed server protection, SLB	21
failover	
active-active	291
active-standby	289
overview	280
fault tolerance	
port trunking	342
Server Load Balancing	28

filtering

allow	53
configuration example	57
default filter	54, 57
deny	53
inserting	54
IP address ranges	69, 87
NAT configuration example	66 to 68
numbering	54
order of precedence	53
proto (option)	57, 82
security example	56
Web OS 8.3 additions	70
firewalls	56
fragmenting jumbo frames	325, 327
frame processing	323
frame tagging. <i>See</i> VLANs tagging.	
FTP Server Load Balancing	44

G

gateway. *See* default gateway.

Global SLB

configuration tutorial	150 to 160
Distributed Site State Protocol	146, 151
DNS resolution (diagram)	147
domain name configuration	155
health check interval	154
hostname configuration	155
HTTP redirect	148
port states	153
real server groups	152
real servers	152
remote site configuration	154
tests	163

H

half-duplex for jumbo frames	323
hash metric	35
health checks	81, 256
configuration using scripts	260
format	259
Global SLB interval	154
hostname for HTTP content	255 to 256
HTTPS/SSL	263
ICMP	252
IMAP server	256
RADIUS server	257
real server parameters	251
script-based	258 to 262
TCP	252
verifying scripts	262
wireless session protocol	252 to 254
hostname, for HTTP health checks	155, 255
hot-standby redundancy	282
configuration	306
HTTP	
application health checks	255
redirects (Global SLB option)	148
HTTPS/SSL health checks	263

I

ICMP	52
IEEE 802.1Q VLAN tagging	319
IF. <i>See</i> IP interfaces.	
IGMP	52
IMAP server health checks	256
imask	33
ingress traffic	125
inserting filters	54
internal routing	333
Internet Service Provider (ISP), SLB example	27
inter-switch port states for hot-standby	284

IP address

conservation	66
filter ranges	69, 87
local route cache ranges	332
private	66
proxies	24, 39, 78, 81 to 84
real server groups	30, 152, 176, 300
real servers	23, 29, 152
routing example	328
SLB real servers	30
virtual servers	23, 24, 31, 153

IP interfaces

configuration example	30, 151
example configuration	328, 331
routing	325
VLAN #1 (default)	318
VLANs	318

IP proxies

for application redirection	84
for Global Server Load Balancing	161
for Server Load Balancing	39
<i>See also</i> proxies, proxy IP address (PIP).	

IP routing	25
cross-subnet example	325
default gateway configuration	329
IP interface configuration	328, 331
IP interfaces	325
IP subnets	326
network diagram	326
routing between VLANs	324
subnet configuration example	328
switch-based topology	327
IP subnets	325, 327
routing	325, 326, 327
VLANs	317, 319
ISL Trunking	341

J**jumbo frames**

ACEnic adapters	323
fragmenting to normal size	325, 327
frame size	323
isolating with VLANs	323
routing	325, 327
supported duplex modes	323
VLANs	323

L

Layer 4	
administrator account	270
optional software	21
least connections (SLB Real Server metric)	35
lmask (local route cache parameter)	332
lnet (local route cache parameter)	332
local route cache address range	332
local route cache parameters	
lmask	332
lnet	332
log (filtering option)	51, 56
logical segment. <i>See</i> IP subnets.	

M

MAC address	127
Management Processor (MP)	318
use in switch security	265
manual style conventions	18
mapping ports	85
mapping virtual ports to real ports	37
maxcons limit	37
maximum connections	36, 37
mcon (maximum connections)	36
minimum misses (SLB real server metric)	34
MP (Management Processor)	318
multi-links between switches	
using port trunking	341
using VLANs	321

N

name servers, Global SLB configuration example ..	147
NAT. <i>See</i> Network Address Translation.	
Network Address Translation (NAT)	51, 81
configuration example	66 to 68
filter example	67
proxy	67
static example	68
network performance stats with proxy addresses ..	39
NFS server	27
non-cacheable sites in application redirection	86
none (port processing mode) example	31
non-HTTP redirects for GSLB	161

O

optional software	77, 145
Layer 4 SLB support	21
OSPF	52
overflow servers	37

P

parallel links	321
passive cookie mode	206
password	
administrator account	270
default	270
L4 administrator account	270
user account	270
PDU's	321
persistence	
cookie-based	204 to 217
SSL session ID-based	218 to 220
persistent bindings	24
PIP. <i>See</i> proxies, proxy IP address.	
port 80	151
port mapping	85
port processing mode	
client	31
none	31
server	25, 31
port states	153
port trunking	342
configuration example	343
description	344
EtherChannel	341
fault tolerance	342
ports	
for services	52
mapping	43
physical. <i>See</i> switch ports.	
SLB configuration example	31
precedence, in bandwidth classifications	229
private IP address	66
private network	66
protocol types	52
proxies	24, 39, 78 to 84
configuration example	67
NAT	66
proxy IP address (PIP)	24, 39, 42, 84
proxy servers	78
PVID (port VLAN ID)	317

R**RADIUS**

authentication	268
authentication for secured switch management	268
health checks	257
server parameters	257
SSH/SCP	275
real server groups	
backup/overflow servers	37
configuration example	30, 152, 176, 300
real servers	24
backup/overflow servers	37
configuration example	152
connection timeouts	36
health checks	251
maximum connections	36
SLB configuration example	30
weights	36
redirect (HTTP)	148
redirection. <i>See</i> application redirection	
redundancy	
active-active	282
active-standby	281
hot-standby	282
remote (Global SLB real server property)	154
roundrobin (SLB metric)	35
routers	326, 329
border	333
peer	333
port trunking	341
switch-based routing topology	327
using redirection to reduce Internet congestion ..	77
web-cache redirection example	78
routes, advertising	333
routing	333
rport (filtering)	81, 85
RSA keys	275

S

scalability, service	21
SCP	272
services	274
script-based health checks	258 to 262
Secure Copy	272
Secure Shell	272
SecurID	276

security

filtering	51, 56
firewalls	56
private networks	66
switch management	265
VLANs	317
segments. <i>See</i> IP subnets.	
server (port processing mode) example	31
server failure	264
Server Load Balancing	
across subnets	325
active-active redundancy	298
backup servers	37
complex network topologies	39
configuration example	27, 39
direct real server access	42
distributed sites	145
failed server protection	21
fault tolerance	28
FTP	44
health checks	251
maximum connections	36
overflow servers	37
overview	22
persistent bindings	24
port processing modes	25, 31
proxies	24, 39
proxy IP addresses	42
real server group	30, 176, 300
real server IP address (RIP)	23
real servers	24
remote sites	145
topology considerations	24
virtual IP address (VIP)	23, 24
virtual servers	23, 31
weights	36
server pool	21
server port processing	25
service failure	264
service ports	52
shared services	21
SIP (source IP address) for filtering	69, 87
smask (source mask) for filtering	69, 87
Spanning-Tree Protocol	
eliminating bridge loops with VRRP	310
replacing with hot-standby failover	306
VLANs	318, 321
spoofing, prevention of	265
sport (filtering option)	57, 82

SSH	272
RSA host and server keys	275
SSH/SCP	
configuring	277
supported client commands	278
static NAT	68
statistical load distribution	342
STP bridge PDUs	321
switch failover	280
switch management	
security	265
via IP interface	318
switch ports VLANs membership	317
synchronization	
configuration	285
VRRP active active mode	308
syslog messages	51

T

tagging. <i>See</i> VLANs tagging.	
TCP	52, 59, 60
health checking using	33
port 80	43
TCP/UDP port numbers	37
TDT (Theoretical Departure Times)	227
Telnet	56
text conventions	18
Theoretical Departure Times (TDT)	227
timeouts for real server connections	36
TOS burst limit for bandwidth management	234
tracking (VRRP)	292
transparent proxies	39, 78, 81 to 84
tunneling	125
typographic conventions	18

U

UDP	52, 59, 60
datagrams	324
jumbo frame traffic fragmentation	327
server status using	33
URL-based	230 to 231
user account	270

V

virtual clocks	227
virtual interface router (VIR)	286
virtual IP address (VIP)	23, 24

Virtual Local Area Networks. <i>See</i> VLANs.	
virtual port mapping to multiple real ports	46 to 49
virtual router group	283
virtual router ID numbering	311
virtual server router	290
virtual servers	23
configuration example	31
IP address	31, 153
VLANs	79
ACEnic adapter support for	319
broadcast domains	317, 319, 321, 330
default	317
example showing multiple VLANs	319
ID numbers	317
IP interface configuration	331
IP interfaces	318
isolating jumbo frames	323
jumbo frames	323
Management Processor	318
multiple links	321
multiple VLANs	318, 319
parallel links example	321
port configuration	330
port members	317
PVID	317
routing	330
security	317
Spanning-Tree Protocol	318, 321
tagging	317 to 320
topologies	319
VLAN #1 (default)	80, 151, 317 to 319
VPN	125
VPN cluster	126
VPN Load Balancing overview	125
VRRP (Virtual Router Redundancy Protocol)	
active-active redundancy	282
active-standby redundancy	281
hot-standby redundancy	282
inter-switch port states	284
overview	286, 292
synchronization	308
synchronizing configurations	284
virtual interface router	286
virtual router ID numbering	311
virtual server router	290
vrid	286

W

web hosting 27

web-cache redirection. *See* application redirection

web-cache servers 77 to 79, 80

weights..... 36

World Wide Web, client security for browsing..... 56

WSP health checks 252 to 254

